UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Arqueología informática: diseño e implementación de calculadoras mecánicas Facit con Scratch

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Christian Calderón Orellana

Tutor: Xavier Molero Prieto

Curso 2019-2020

Resum

El present treball efectua una investigació històrica, deductiva i pràctica sobre les calculadores mecàniques, especialment en els models de meitats del segle XX (mecanismes Odhner). En la primera part s'exposa una anàlisi d'índole històric respecte al passat de les eines de càlcul, enfocat en el desenvolupament i evolució des del seu origen fins al seu màxim auge, centrant-se amb mes detall en la companyia Facit i dues dels seus models mes importants (Facit C1-13, Facit CA1-13). En segon lloc es presenta l'elaboració de dos projectes desenvolupats amb el llenguatge de programació Scratch, que simulen el comportament dels models descrits. Finalment, per mitjà de la pàgina web del Museu d'història informàtica de la Universitat Politècnica de València es compartirà els resultats obtinguts.

Paraules clau: Facit, Scratch, història del càlcul

Resumen

El presente trabajo efectúa una investigación histórica, deductiva y práctica sobre las calculadoras mecánicas, especialmente en los modelos de mitades del siglo XX (mecanismos Odhner). En la primera parte se expone un análisis de índole histórico con respecto al pasado de las herramientas de cálculo, enfocado en el desarrollo y evolución desde su origen hasta su máximo auge, centrándose con mas detalle en la compañía Facit y dos de sus modelos mas importantes (Facit C1-13, Facit CA1-13). En segundo lugar se presenta la elaboración de dos proyectos desarrollados con el lenguaje de programación Scratch, que simulan el comportamiento de los modelos descritos. Finalmente, por medio de la página web del Museo de historia informática de la Universidad Politécnica de Valencia se compartirá los resultados obtenidos.

Palabras clave: Facit, Scratch, historia del cálculo

Abstract

This work carries out historical, deductive and practical research on mechanical calculators, especially in the mid-20th century models (Odhner mechanisms). The first part presents an analysis of a historical nature with respect to the past of the calculation tools, focused on the development and evolution from their origin to their maximum boom, focusing in more detail on the company Facit and two of its most important models (Facit C1-13, Facit CA1-13). In second place, it is presented the elaboration of two projects developed with the programming language Scratch, which simulate the behavior of the described models. Finally, the results obtained will be shared through the website of the Museum of Computer History of the Polytechnic University of Valencia.

Key words: Facit, Scratch, history of calculus

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

El primer capítulo muestra brevemente el trabajo, la motivación que lo originó, los objetivos a alcanzar, la estructura que comprende y una concisa descripción acerca de la bibliografía.

1.1 Motivación

Comprender el presente es evitar ignorar el pasado, por este motivo la arqueología informática pretende enseñar a través de diversas disciplinas (cálculo, informática, historia...) el pasado interior de la informática. La explicación del marco histórico en el desarrollo y evolución de las diferentes herramientas de cálculo es una manera de conocer la historia de la informática.

Existen muchas maneras de comprender el pasado de las calculadoras, ya sea por documentos antiguos (manuales, patentes) o pruebas históricas (herramientas de cálculo descubiertas). Sin embargo, hoy en día gracias a las tecnologías de la información y comunicación es viable emular y compartir un proyecto exacto de calculadora mecánica, pudiendo ser accesible a todo el mundo desde el confort de un navegador web (Sitio web del Museo de informática), continuando ese compromiso con la sociedad (difusión patrimonial, enseñanza, estudio y recreo) que la Universidad Politécnica de Valencia ha generado desde la creación del museo y de esta manera cumplir varios Objetivos de desarrollo sostenible (ODS).

El trabajo se centra en las calculadoras mecánicas Facit (tipo Odhner). Este tipo de calculadora es uno de los antecesores a las calculadoras digitales actuales como herramienta de cálculo aritmético. Por esa razón, es importante conocer la naturalidad histórica de un antecesor próximo a la calculadora moderna, comparando objetivamente su desempeño, funcionalidad y límites.

1.2 Objetivos

El objetivo del trabajo es mostrar una perspectiva universal con respecto a la historia de las herramientas de cálculo, exponiendo el origen y evolución a lo largo del tiempo, focalizando con mayor detenimiento en las calculadoras Facit (tipo

Odhner). Asimismo de presentar dos proyectos de simulador para incorporar a la web del Museo de Informática de la Escuela.

Con mas detalle, estos son los principales objetivos para este trabajo:

1. Ofrecer una percepción universal de la evolución histórica de las diversas herramientas de cálculo a través del tiempo.
2. Descubrir la historia de la compañía Facit, así como todos sus principales modelos desarrollados.
3. Conocer sus principales innovaciones mecánicas implantadas por la compañía Facit en sus modelos.
4. Entender el funcionamiento de algunos modelos de calculadoras Facit (C1-13, CA1-13), enseñando sus técnicas de uso.
5. Llevar a cabo en Scratch dos simuladores que imiten el funcionamiento de los modelos estudiados.

1.3 Estructura de la memoria

El trabajo contiene una estructura de 8 capítulos, incluyendo un apéndice. Seguidamente se muestra de modo detallado y resumido cada uno de los capítulos que forman parte de este documento:

Capítulo 1. Capítulo inicial donde se expone la motivación, los objetivos planteados, estructura del documento y notas sobre la bibliografía.

Capítulo 2. En este apartado se analiza el contexto histórico acerca del origen, desarrollo y evolución de las diferentes herramientas de cálculo. Además se muestra una introducción sobre el sistema mecánico que la compañía Facit usó para su producción de calculadoras.

Capítulo 3. Se describe concretamente la historia de la compañía Facit, indicando la evolución de sus modelos a partir de su innovación. Además se muestra información sobre los modelos desarrollados en forma de tablas.

Capítulo 4. El capítulo estudia la utilización de las calculadoras *C1-13* y *CA1-13* mediante un resumen de sus principales manuales Facit.

Capítulo 5. Realizar un recorrido sobre las principales aportaciones que ofrece el lenguaje de programación Scratch. A su vez, realiza una introducción sobre su entorno, componentes y bloques de programación.

Capítulo 6. Después de conocer la historia, funcionamiento y mecanismos de los proyectos. En este capítulo se muestra el análisis, diseño e implementación de ambas calculadoras.

Capítulo 7. Detallar el Museo de Informática de la Universidad Politécnica de Valencia y la página web que se desarrolla sobre el tema planteado.

Capítulo 8. Por último, se explica los objetivos cumplidos del trabajo. Además de una descripción acerca de las posibles ampliaciones sobre los proyectos.

1.4 Notas sobre la bibliografía

Para desarrollar el presente trabajo fue necesario la utilización de material de apoyo, siendo consultadas y después colocadas en la bibliografía final. Fue necesario realizar consultas en libros, artículos de revistas científicas y páginas web especializadas en inglés, español, sueco y alemán. Además se utilizó varios manuales Facit para el estudio de cada una de las calculadoras planteadas, con el fin de entender su funcionalidad para después ser implementadas.

Para el contexto histórico y la descripción de cada herramienta de cálculo mostrada en el capítulo, fue esencial la utilización de obras generales tales como, la Historia universal de las cifras de Georges Ifrah, *History of computing technology* de Michael R. Williams y el libro Historia de las Matemáticas de K. Ríbnikov[11, 12, 10]. Además para temas más concretos se usaron las siguientes referencias [1, 2, 3, 8, 9, 14, 15, 16, 17], y con respecto a la descripción de la mecánica (*pinwheel*) utilizada por la compañía Facit, se hizo uso de la siguiente referencia[18].

El capítulo dos al ser una descripción mas específica y detallada sobre la compañía Facit, era necesario usar artículos divulgativos realizados por universidades originarias del lugar de la extinta compañía (Suecia), en donde se encontró la siguiente referencia[4], y para la parte histórica y técnica de cada una de las calculadoras desarrolladas por la compañía Facit, se encontró las siguientes bibliografías de apoyo [19, 20]. El número de calculadoras creadas por la compañía eran relativamente extensas y variadas, por ese motivo hizo necesario la representación de los distintos modelos en forma de tablas, donde cada conjunto específico tiene las características generales y patentes de cada modelo calculador (tiene un hipervínculo a la web de la patente). Ver Tablas 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7.

Sobre el funcionamiento, manejo y técnicas para realizar cálculos en las calculadoras Facit *C1-13*[1] y *CA1-13*[2], fue necesario estudiar sus principales manuales de uso para cada calculadora. Los manuales Facit tienen una descripción general sobre cada modelo, detallando sus partes principales, y a su vez muestra con ejemplos prácticos el uso de la calculadora. Hay que señalar que el manual *CA1-13* tiene mejor descrita los pasos para la realización de operaciones aritméticas. Aparte de los manuales, fue de gran ayuda tocar de primera mano simuladores referentes a los modelos que se estaban estudiando. Estos simuladores encontrados implementados por otros desarrolladores tienen acceso público vía web[3].

Para la descripción de la plataforma Scratch se llego a consultar una variedad de material de apoyo como por ejemplo: manuales, libros y artículos investigativos sobre Scratch. Entre el material tenemos el artículo de Jesús Moreno, Gregorio Robles y Marcos Román sobre las razones para fomentar el «Pensamiento Computacional»[5], y también el artículo de Resnick Mitchell [6] sobre el contexto histórico y analítico del lenguaje de programación Scratch. Además para el aprendizaje del uso de la plataforma se utilizaron las siguientes referencias [13, 21].

Por último tenemos el material consultado acerca a la divulgación patrimonial que el Museo de Informática realiza y aporta a la sociedad [7].

[1] Manual disponible en inglés: https://url2.cl/g2Vsw
[2] Manual disponible en alemán: https://www.manualscat.com/es/facit-ca1-13-manual
[3] Simuladores encontrados en: http://www.rechenautomat.de/Facit/Rechner.php

CAPÍTULO 2
Contexto histórico

Los obstáculos, dificultades y desafíos hacen despertar el ingenio para poder crear o desarrollar instrumentos que ayuden a mejorar tareas que puedan resultar complejas o imposibles, desde la evolución del ser humano como especie hasta la actualidad, este ingenio se refleja en la evolución histórica de los mayores inventos de cálculo creados. Como ejemplos tenemos el ábaco, con el objetivo de efectuar operaciones aritméticas complejas para un mercader, hasta las supercomputadoras capaces de ayudar a científicos con simulaciones en entornos inimaginables. Por ello, como ingenieros informáticos, es interesante hacer hincapié en cómo una herramienta logra realizar cambios en la vida cotidiana de las personas. Adicionalmente, tener conocimiento sobre herramientas de cálculo mecánicas de forma más detallada permitiendo contemplar de manera general una perspectiva a las diferentes rutas que ayudaron al desarrollo actual de los sistemas de cálculo moderno, siendo el efecto de las soluciones ingenieriles y matemáticas durante mucho tiempo.

Por ese motivo, en este capítulo vamos a mostrar una perspectiva general acerca de las herramientas de cálculo, basándose en el origen y evolución de estas herramientas. Empezamos con su contexto histórico a través del tiempo, ofreciendo una comprensión sobre el pasado del cálculo hasta llegar al origen de las primeras calculadoras mecánicas. Para finalmente exponer tres tipos de sistemas mecánicanicos de cálculo que revolucionaron la historia, y a su vez dar una introducción histórica y mecánica al origen de las calculadoras Facit.

2.1 Primeras herramientas de cálculo

La sabiduría y la práctica siempre han sido uniformes a lo largo de la historia de la Humanidad. Saber desarrollar aquellas en una utilidad técnica, hace trascender con mayor fuerza la enseñanza al ser humano frente a problemas de la vida cotidiana. Este «ingenio» se puede apreciar en la prehistoria, muy concretamente en el Paleolítico Superior (hace 35.000 años), donde huesos o piedras de forma alargada y con muescas eran usados para representar una serie numérica de forma arbitraria y fija [8, 12]. En consecuencia, debido a su simplicidad hizo factible realizar los primeros registros numéricos y cálculos matemáticos, ayudando más adelante a desarrollar métodos de cálculo mucho más comple-

ja logrando ampliar una mayor magnitud numérica de centenas a miles y más tarde a millones [8].

En la Figura 2.1 tenemos un ejemplar de hueso con muescas, conocido como "Hueso de Ishango", hallado en 1960 por el geólogo belga Jean de Heinzelin de Braucourt en el área africana de Ishango, cerca del nacimiento del río Nilo. Este primitivo instrumento de forma alargada contiene un pequeño trozo de cuarzo inamovible, ubicado en el lado extremo izquierdo. Este cuarzo podría haberse usado para hacer tallas o incisiones [8, 11].

Figura 2.1: Seis vistas diferentes del hueso de Ishango. Imagen oficial del Real Instituto Belga de Ciencias Naturales

En el Antiguo Egipto fue de vital importancia el cultivo de las matemáticas y el cálculo. El poder llevar una organización del número mínimo de las cosechas, gestionar los impuestos, predecir los ingresos y gastos son algunas de las inquietudes elementales. Por ese motivo, los egipcios muestran soluciones innovadoras, prácticas y parcialmente sencillas con fórmulas de cálculo de series numéricas [9].

Los papiros matemáticos de esta civilización son la primera prueba de la existencia del cálculo aplicado a la resolución de problemas, definiendo un sistema de numeración "jeroglífico-decimal". Con ayuda de este sistema, los egipcios eran capaces de llevar a cabo todos sus cálculos con números enteros, desarrollando métodos y soluciones comunes que eran descritos en el documento. Actualmente hay dos existencias descubiertas de papiros matemáticos que datan del año 2000 a.c, uno se encuentra en Londres "Papiro de Rhind" (llamado así por el científico que lo descubrió) y el segundo en Moscú [9, 10]. Como se puede observar en la Figura 2.2 el sistema de números egipcio está basado en la base 10 donde los números claves de la forma 10^k (k = 0, 1, 2, 3, 4, 5, 6, 7) es representado por jeroglíficos individuales, dando lugar a una representación más amplia de las cantidades numéricas [10, 12].

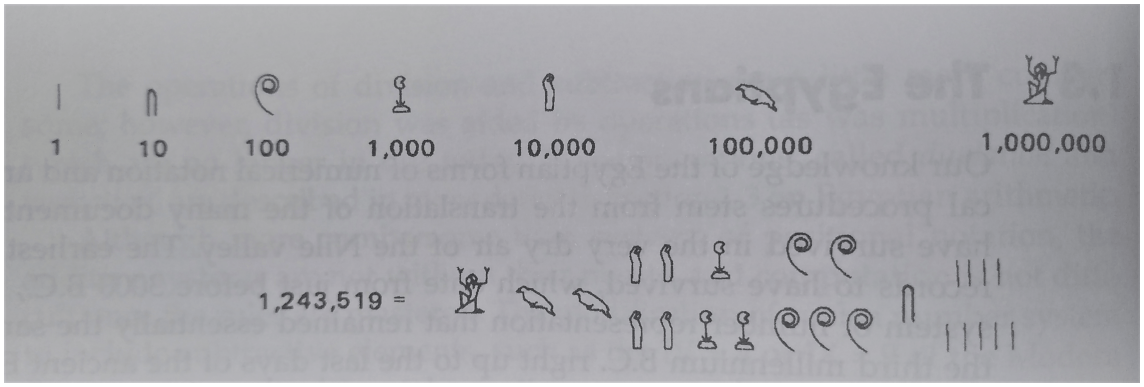

Figura 2.2: Jeroglíficos egipcios con su respectiva representación numérica. Ejemplo de una cantidad de siete cifras.

La evolución del sistema numérico continuó su curso en las distintas civilizaciones de la era (Egipcia, Griega, Romana, China) basándose en un sistema de símbolos en base 10, siendo el sistema babilonio de símbolos matemáticos el único en diferenciarse (sistema sexagesimal de base 60) [10, 11]. Todos estos sistemas tienen una particularidad de origen común y es gracias a la anatomía humana, por ejemplo, los números 1, 10, 20 y 60. El 1 tiene singularidad al único dedo grande de la mano (dedo pulgar), el 10 es el número total de los dedos de las dos manos, el 20 es resultado de contar los dedos de todas las extremidades. Y por último el 60 es el resultado de contar 5 veces en una mano el número de docenas, es decir, sumar cinco veces 12 con la mano.

Hoy en día no existe ninguna prueba física sobre el origen y la etapa exacta del contar con los dedos, siendo muy probable que se remonte desde la época de la prehistoria. De hecho, las únicas pruebas han sido las obras de Herodoto (484-425 a.c), relatando el uso de este sistema por los griegos en el siglo V a.c , un comentario hecho por el político romano Cicerón (106-3 a.c.) hacia su amigo Ático para debatir la distinción sobre los dos tipos de interés (simple y compuesto) y una mención en la obra "La ciudad de Dios" a San Agustín (354-430) exponiendo a sus oyentes el modo de entender algún sistema de cálculo con los dedos, refiriéndose a números que se mantienen en la mano y ocasionalmente pidiendo a la audiencia que siga algún cálculo particularmente complejo moviendo sus dedos mientras expone sobre algún tema oscuro cabalista [12, 11].

La evolución de los sistemas numéricos y técnicas de cálculo fue de gran ayuda para la aparición de instrumentos avanzados. El «ábaco», palabra latina con raíces de terminología griega Abax (tabla en griego), es el precursor de las distintas herramientas de cálculo existentes en la actualidad. Y es que el origen de esta herramienta puede estar apuntando desde la antigua Babilonia gracias a la tabla de contar más antigua encontrada "La tablilla de Salamina" (300 a.c), descubierta en 1846 en la isla de Salamina [14]. El desarrollo de este instrumento fue mejorándose a causa del intercambio cultural y comercial con Grecia, llegando más adelante también a Roma. Los romanos, debido a la necesidad de calcular números muy complicados y extensos, se vieron obligados a desarrollar una versión portable, robusta y práctica de la herramienta siendo la versión más costosa para su época "El abaco de mano romano", mírese la Figura 2.3.

Este instrumento forjado en metal o bronce contiene siete hendiduras cortas y siete largas usadas para realizar cálculos con números enteros; la parte derecha de la tablilla posee tres hendiduras que son usadas para cálculos fraccionarios. Los símbolos tallados en cada una de las columnas muestran las distintas cifras del sistema numérico romano, pudiéndose leer de derecha a izquierda las unidades (I), las decenas (X) y las centenas (C). Las hendiduras superiores contienen una única canica, en cambio las hendiduras inferiores contienen cuatro canicas, salvo las dos columnas de la derecha, marcadas como 2 y θ o las tres hendiduras separadas con 3, S o ε. Para las matemáticas de base mixta se utiliza la hendidura que está en la parte central marcada con una C hacia atrás y la columna con el símbolo 2 al lado de la ranura inferior [14].

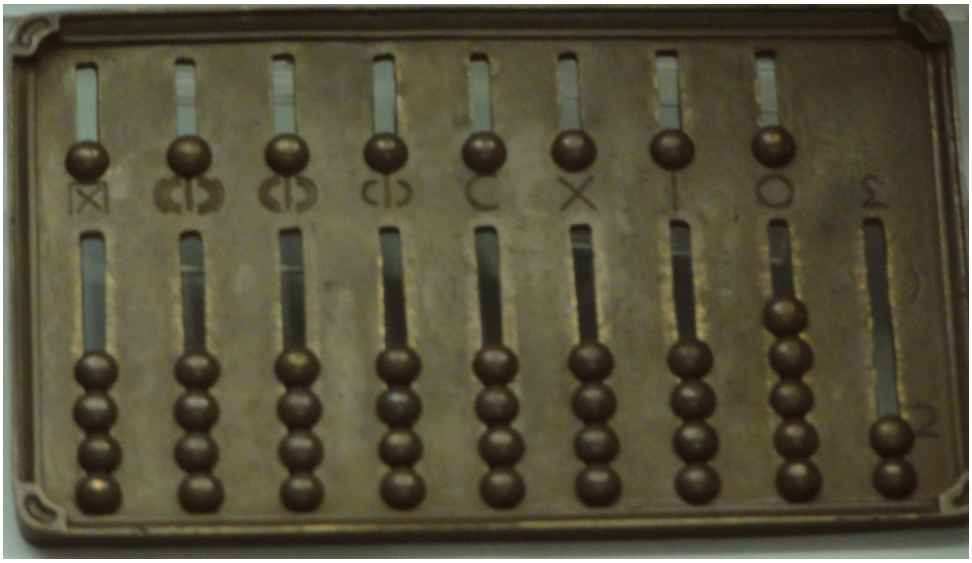

Figura 2.3: Réplica del ábaco romano de bolsillo expuesto en el Museo de Ciencias de Londres. Tiene un tamaño de 12x8 cm.

Durante toda la Edad Media los conocimientos matemáticos en Europa eran muy escasos y no se conocen descubrimientos destacables. Las personas instruidas eran minoritarias debido a que la Iglesia tenía el control y acceso a estos conocimientos. Los únicos en conservar esta sabiduría fueron los monjes científicos, manteniendo, estudiando y copiando obras de personajes antiguos importantes. En esta época surgieron los primeros centros especializados tal como "La escuela de Gerberto" (940-1003) creado y organizado en la ciudad de Reims (Francia) por Gerberto de Aurillac, un monje destacado en las matemáticas convirtiéndose más adelante en el papa romano Silvestre II que ayudó a introducir en Europa el sistema arábigo de números [10, 1].

A parte de otras ciencias la escuela de Gerberto instruía en cálculo usando una versión mejorada del ábaco. En la Figura 2.4 se muestra el ábaco de Gerberto conteniendo 27 posiciones, donde 9 fichas son colocadas con los números arábigos grabados. Se interpreta de derecha a izquierda, es decir, la primera columna del extremo derecho incluye las unidades; la segunda columna, las decenas; la tercera columna, las centenas, y así sucesivamente [12].

Con este ingenioso y poco popular ábaco, multiplicar y dividir resultaba fácil y sencillo, dejando asentadas las bases para que más adelante, otros eruditos mejorasen el sistema con la introducción del cero[1].

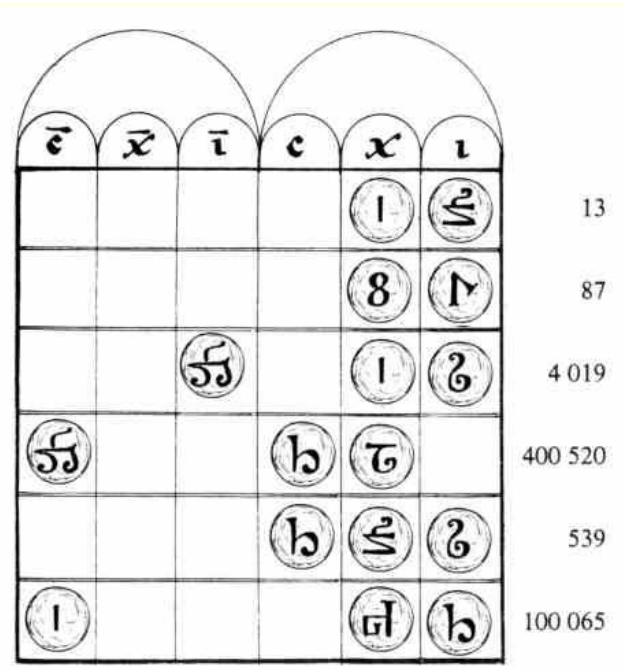

Figura 2.4: Parte del manuscrito del siglo XI encontrado en Limoges (Francia 1030). Muestra el ábaco de Gerberto con varias representaciones numéricas, pudiéndose observar la ausencia del cero en la tabla. La privación de un símbolo en el marcador significa de forma implícita el número cero.

2.2 Evolución de los sistemas mecánicos de cálculo

En la enseñanza de la Europa medieval existían muchos sistemas de cálculo, desarrollándose durante este tiempo dos rivalidades: los «abaquistas» y los «algorítmicos». Los primeros, se diferenciaban en el uso indispensable del ábaco y la numeración romana. En cambio los algorítmicos utilizaban notaciones escritas de cifras hindúes aplicando este sistema numérico, incluso añadiendo el signo cero en alguno de ellos, realizando el cálculo en papel sin necesidad de un instrumento. En la disputa se formaron los sistemas de numeración y métodos de cálculo aritmético, cada vez más próximos a los sistemas y técnicas comunes actuales [10].

Durante más de tres siglos existieron conflictos teológicos ligados a las matemáticas: según la interpretación de las Escrituras la nada o el vacío no podía existir en el origen, dando como resultado duros debates para la incorporación del cero al sistema numérico decimal. Por ese motivo, aparece en el año 1202 el libro "Liber abaci, el Libro del ábaco" publicado por Leonardo de Pisa (conocido como Fibonacci), ayudando al entendimiento de lo racional al conflicto entre

[1] Cosas a las que la religión se opuso por demoníacas o herejes: el cero. Consultado en https://www.ateoyagnostico.com/?p=10281

religiosos católicos y matemáticos orientales con el número cero (zephirum, una latinización del árabe sifr). El libro es una enciclopedia de los conocimientos matemáticos usados por todos los pueblos del mar Mediterráneo. Después de su salida hasta más de 200 años, originó un modelo inmejorable de obras matemáticas para Europa y proyectó nuevos logros matemáticos en la época del Renacimiento [10, 11].

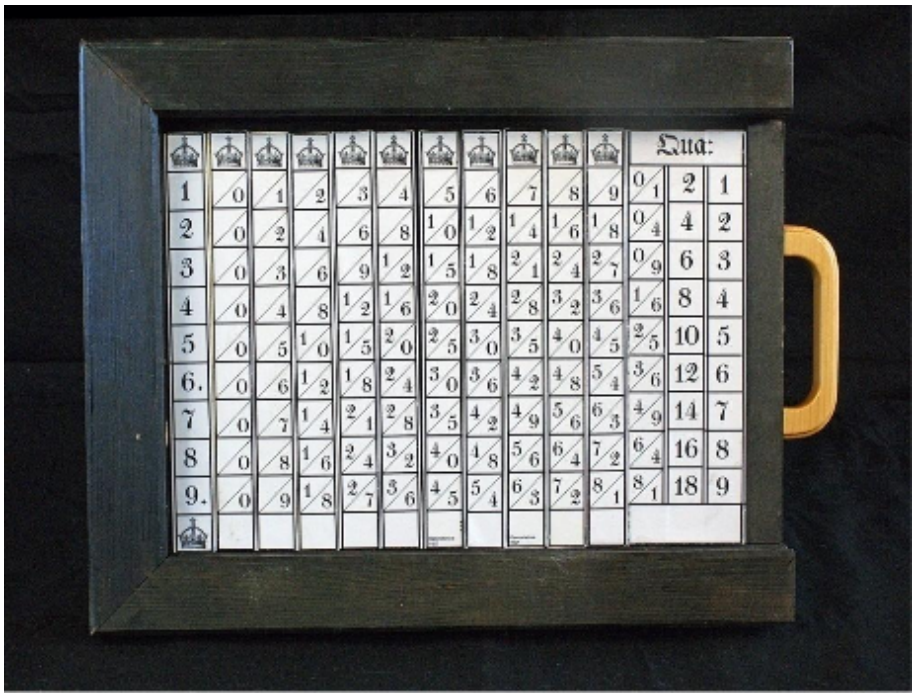

Figura 2.5: Réplica de las varillas de Napier. Construida por F. Massen para la exposición Çalculs àl'Ancienne"(MNHN, Cirque des Sciences, 2007).

A principios del Renacimiento surgió la innovación en todas las ciencias especialmente las matemáticas. En aquel tiempo, los cálculos realizados para la astronomía llegaban a ser terriblemente extensos, siendo necesario una herramienta para agilizar este proceso. Las varillas de Napier, conocidas también como «Huesos de Napier», ofrecieron la ventaja de simplificar cálculos numéricos engorrosos a los astrónomos de la época, aplicándose poco después a otras ciencias. En 1617 el matemático John Napier, desarrolla un sistema semiautomático de cálculo basado en una tabla de multiplicar con elementos móviles. Con este sistema se transforma multiplicaciones en simples sumas. En la Figura 2.5 se puede observar una réplica de las varillas de Napier, donde cada pieza (base cuadrada) esta formada por un centímetro de lado, y de 10 cm de largo, con forma prismática hecho de madera o marfil. Además cada pieza viene acompañado de grabados en cada una de sus caras, simbolizando la unidad y sus múltiplos, siendo un conjunto total de 10 varillas, enmarcados entre el 0 y el 9 [12, 11].

Poco después de la invención de Napier, el matemático alemán Wilhelm Schickardt, gran amigo de Kelpler, desarrollaría en 1623 la primera máquina calculadora. El Reloj Calculador (Rechenuhr) es considerado el primer sistema calculador, donde utiliza un complejo sistema de engranajes y varillas mecanizadas permitiendo hacer operaciones de forma mecánica. El sistema mecánico de cálculo incluye el principio de las regletas de John Napier. Gracias a la relación que tenía con Kepler los apuntes y bocetos de la máquina se salvaron, sabiéndose que el

inventor desarrolló una única máquina siendo destruida poco después de un año por un incendio. Con los mapas de la máquina encontrados se realizaron varias réplicas exactas de la calculadora.

En la Figura 2.6 tenemos una réplica exacta del Reloj Calculador. La máquina es capaz de realizar las cuatro operaciones aritméticas empleando acarreos y números de seis dígitos. El mecanismo de engranajes y cilindros junto a las varillas de Napier ejecutaban las multiplicaciones y divisiones de forma semiautomática. En cambio, las operaciones de suma y resta no necesitaban de la intervención del usuario, ya que solo era necesario introducir los números para que el sistema lo realice automáticamente.

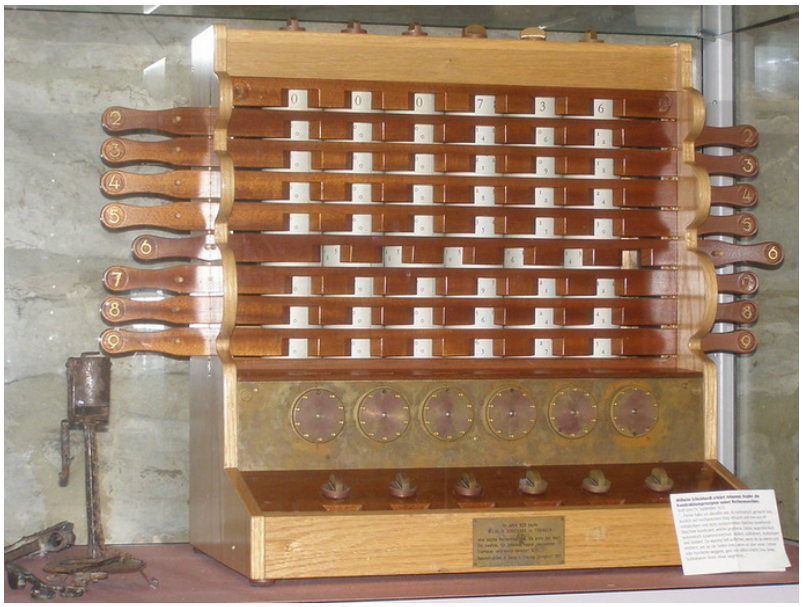

Figura 2.6: Réplica del Reloj Calculador de Wilhelm Schickardt expuesto en el museo Herrenberg (Alemania). Considerado como el primer dispositivo calculador mecánico.

Para los siguientes años, el inventor y matemático francés Blaise Pascal crearía en 1642 una maquina automática de cálculo llamada «Pascalina». El invento viene de la necesidad de efectuar operaciones extensas y complejas que su padre Étienne Pascal realizaba como recaudador de impuestos. El joven con 19 años desarrolla su primera versión con cinco ruedas, agregando funcionalidades aritméticas como suma y resta (además ayuda a realizar multiplicaciones y divisiones). La Pascalina tuvo éxito entre familiares y amistades del propio inventor, despertando una idea de comercialización del producto, construyendo más de cincuenta modelos. Estos primeros modelos fueron mejorados en resistencia y solidez del mecanismo (agregó 3 cilindros más al sistema). Sin embargo, el producto no tuvo el éxito comercial deseado, debido a que su elaboración artesanal y material eran costosos, siendo poco rentable al consumidor final por su alto precio de venta, convirtiéndose en una máquina muy exclusiva para pocas personas. Por esta razón, los dueños las utilizaban como herramientas personales para su casa y no para la oficina [12, 2].

Como se observa en la Figura 2.7 el artefacto es una caja rectangular de unos 15 centímetros de ancho y 38 centímetros de largo, teniendo en la cara superior ocho ruedas, utilizadas como marcadores de datos. En el alrededor de cada rueda tiene marcado los números del 0 al 9. En la parte superior de cada rueda es visible

una pequeña ventana mostrando los tambores con las cifras. Cada tambor interior es independiente, conteniendo dos líneas numéricas del 0 al 9 de forma creciente y otra decreciente. Para que el sistema realice operaciones con más de una cifra es necesario ampliar el número de ruedas [2].

A mediados del siglo XVII, un gran académico, inventor y matemático inglés crearía una ligera competencia a las máquinas calculadoras de Pascal. Samuel Morland se acredita como un personaje importante en los primeros desarrollos de la hidráulica, la potencia de vapor y el cálculo matemático. En este último, es muy destacable porque en 1660 ideó un total de tres máquinas de calcular: una para trigonometría (1663), una para sumar y restar (1666) y otra para multiplicar y dividir (1662). Sus máquinas se convirtieron en instrumentos muy populares para fabricantes en Londres y no dejaron de desarrollarlas hasta 1710 [3].

Figura 2.7: La pascalina. Calculadora monetaria francesa expuesta en el Computer Museum History Center (Mountain View, EEUU).

Morland en 1653 fue elegido para acompañar una misión diplomática británica a la corte de la reina de Suecia. La reina Christina fue una notable seguidora de las ciencias y es probable que en su corte Morland se familiarizara con la máquina de Blaise Pascal. Con este importante suceso diez años después comenzó el desarrollo de sus tres máquinas calculadoras. Las máquinas calculadoras de Morland fueron de las más destacadas y diseñadas entre 1662 y 1666. La primera máquina de Morland, llamada máquina de multiplicación, está basada en el principio de las varillas de Napier. El dispositivo es hecho con plata, latón, madera y cristal, teniendo unas dimensiones de 18 centímetros de ancho y 55.5 centímetros de largo. El sistema solo simplifica los productos intermedios, utilizando las barras de Napier para realizar operaciones de forma semiautomática. Su segundo dispositivo desarrollado, es una calculadora de bolsillo, que medía 122 milímetros de largo y 71 milímetros de ancho con una construcción de plata y latón. En la parte superior del dispositivo lleva instalados ocho diales graduados con inscripciones de las escalas en el anillo de su alrededor. Las tres escalas inferiores están comprendidas en 4, 12 y 20 pares, utilizadas para los cálculos con unidades monetarias tales como: centavo, chelín (equivalente a 12 centavos) y guinea (equivalente a 20 centavos). Por último, la parte superior tenemos los 5 diales con

escalas decimales describiendo las unidades, decenas, centenas, miles y décimos. Mírese la Figura 2.8 [12, 3].

Para mediados y finales del siglo XVII el filósofo y matemático alemán Gottfried Leibniz desarrolla el principio mecánico para el cálculo avanzado, soñando con un dispositivo lógico, combinando la aritmética y la lógica, imaginando un instrumento que vaya más allá de la máquina calculadora. Igualmente encontró una solución factible a los procesos computacionales con una codificación de números binarios tal y como lo detalla en sus manuscritos "De progressione Dyadica" (1679) y "Explication de l'Arithmetique Binaire" (1703), describiendo una máquina calculadora que funciona con un sistema binario. La idea innovadora no se llevaría a cabo hasta pasar dos siglos y medio [15].

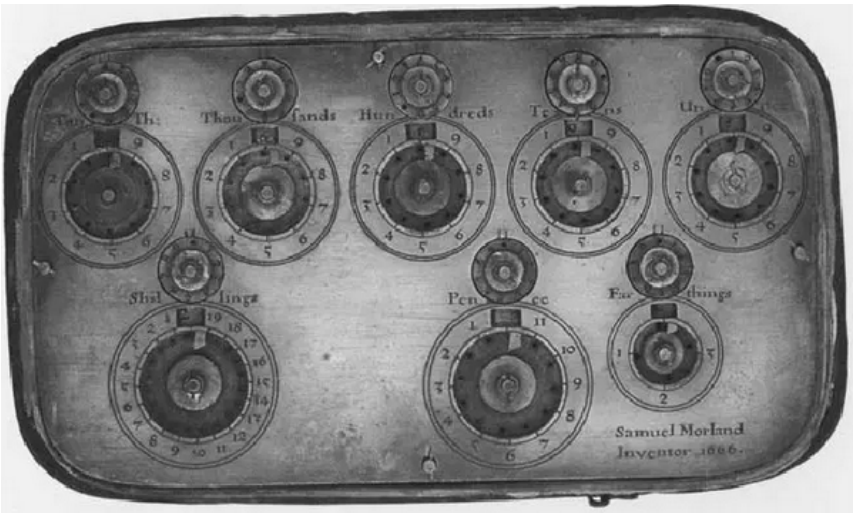

Figura 2.8: Máquina sumadora de bolsillo de Samuel Morland basada en el diseño de la Pascalina.

En 1670 empezó con su diseño de máquina calculadora, llamado "Instrumentum Arithmeticum" (posteriormente lo llamaría "Staffelwalze"). Sin embargo, un modelo de máquina no fue construido hasta 1672 tomando como base la máquina calculadora de Pascal. El sistema que trató de usar era muy similar al de Pascal, dándose cuenta más adelante de las carencias que tiene el sistema al realizar multiplicaciones y divisiones (no es posible ingresar el multiplicando de forma repetida), obligado a desarrollar un mecanismo diferente: "La rueda de Leibniz". Cuando comenzó a crear su primera versión, Leibniz se enfrentó a problemas de producción (mecanismos difíciles de desarrollar, escasos profesionales del sector).

Viéndose en ligeros apuros conoce a M. Olivier, un relojero local de París muy profesional en su oficio que le ayuda a desarrollar el prototipo y los modelos que posteriormente va mejorando. En 1673 Leibniz visitó Londres debido a una misión diplomática, conociendo inventores y científicos como Samuel Morland. Además, realizó demostraciones con su máquina calculadora a la Royal Society, mostrando una gran impresión siendo más tarde invitado para ser miembro del grupo. Desde 1673 hasta 1690 continuó mejorando y demostrando el funcionamiento de la máquina a la Academia inglesa y a la Royal Society, destacando su sistema innovador de cilindro dentado (La rueda de Leibniz) dando otra solución al mecanismo de Pascal y convirtiéndose en el sistema que muchos inventores y

próximamente compañías usarían en sus máquinas calculadoras. En la Figura 2.9 se muestra la última versión desarrollada [12, 15].

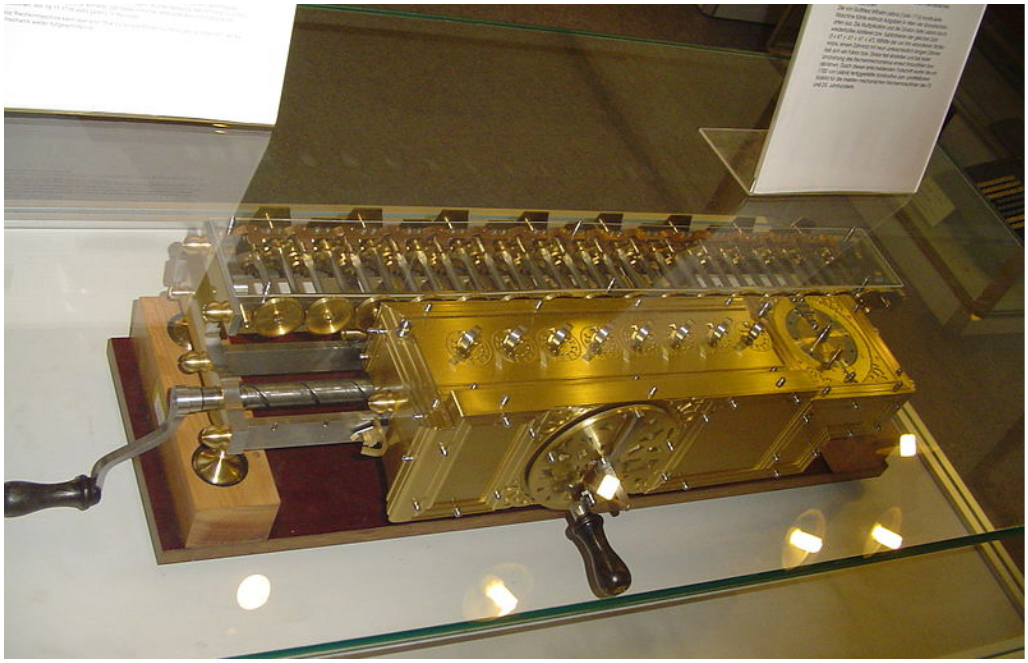

Figura 2.9: Réplica de la calculadora de Leibniz (Stepped Reckoner) exhibida en el museo Technische Sammlungen en Dresden (Alemania).

2.3 Calculadoras mecánicas contemporáneas

2.3.1. La Máquina Analítica

Charles Babbage es uno de los personajes más destacados en la informática y las calculadoras mecánicas debido a sus grandes aportes como matemático e informático teórico creando «La Máquina Analítica». Sus esfuerzos en desarrollar este sistema mecánico de cálculo programable le convirtieron en el pionero de las ciencias computacionales considerándolo por muchos como «el padre de la informática» [12].

Su gran interés por las matemáticas hizo que Babbage realizara cambios en la enseñanza de esta rama, reformando y revitalizando las matemáticas británicas a principios y mitad del siglo XIX. Durante su desarrollo como matemático Babbage tuvo una visión para la construcción de una máquina calculadora. Su desconocimiento en el área mecánica hizo que emprendiera un viaje por Londres y gran parte del continente europeo en vista de los principios de la revolución industrial, los procesos de fabricación y la maquinaria. El interés y descubrimiento de varios hallazgos se vieron reflejados en el libro "On the Economy of Machinery and Manufactures" publicado en 1832, mostrando una visión pionera en economía industrial y procesos operativos que hoy en día continúa [16].

La idea de una máquina calculadora empezó en 1820 y fue desarrollándose hasta la creación de la «Maquina Diferencial» (ver Figura 2.10) en 1822, pero no trató de diseñar un sistema diferente durante más de una década. En todo este

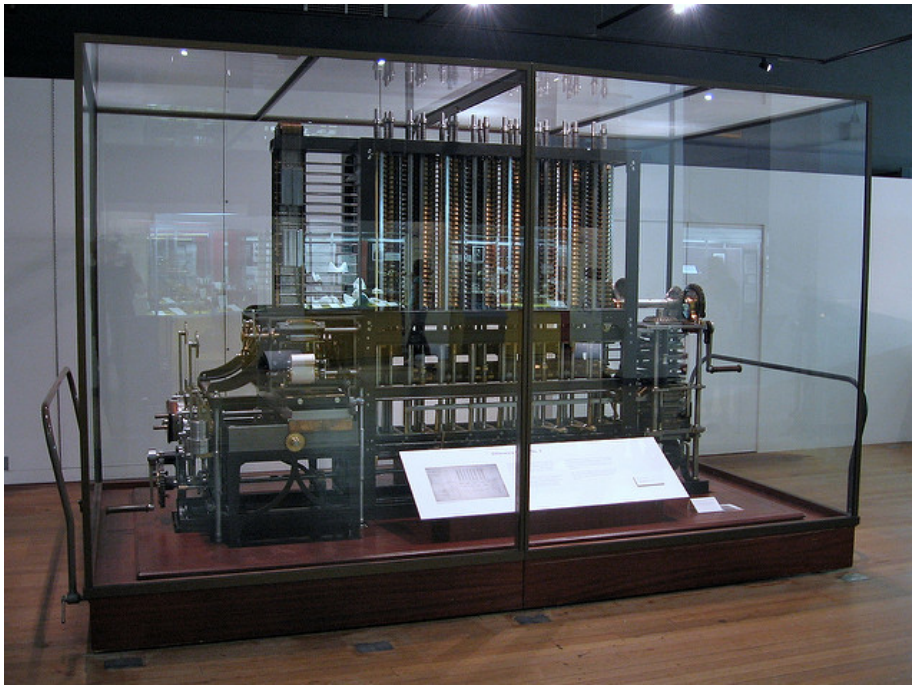

Figura 2.10: Máquina Diferencial de Babbage expuesta en el Museo de Historia de la Ciencia (Londres). Considerada como la primera computadora mecánica de la historia.

tiempo Babbage continuó con su primer proyecto, manteniendo las relaciones de apoyo con el gobierno sobre el futuro de la máquina calculadora. En 1834 realizó varias mejoras, especialmente en la «unidad central de procesamiento» con la que llevaría a cabo las diversas funciones aritméticas. El mecanismo llamado "the mill", al igual que un procesador actual, proporcionaba el almacenamiento de los registros, realizaba las operaciones elementales, controlaba las instrucciones ligadas al usuario y sincronizaba el orden de los procesos. Además de otra sección fundamental, «la memoria» llamada "store" siendo un depósito donde se colocan las cantidades y los resultados de las operaciones guardándolas en columnas numéricas. Para 1835, Babbage decide desarrollar un sistema de «salida de datos», un mecanismo independiente de impresión que tenía una conexión directa con la máquina principal para poder tener una salida numérica impresa a través de papel.

Finalmente, en 1836 tras ver varios defectos en la Máquina Diferencial (el modelo limitaba el control del mecanismo al manejar diferentes y muchas instrucciones a la vez) Babbage descubrió la manera de solucionar el problema aplicando a su modelo «las tarjetas perforadas» como sistema de «entrada de datos», observando el funcionamiento del sistema de control de los telares automáticos mediante el uso de tarjetas perforadas que Joseph Marie Charles había mejorado a principios de siglo. En la práctica, las tarjetas de Babbage tienen que ser detectadas por los mecanismos para introducir datos y órdenes al procesador mecánico. La introducción de este sistema ayudó a controlar mejor el modelo, pudiéndose almacenar los datos y repetir instrucciones, despreocupándose por errores al configurar la máquina de forma manual. A partir de aquí y en adelante Babbage solo se dedicaría a realizar mejoras en el modelo analítico de su máquina agregando funcionalidades aritméticas, mejorando la eficiencia y mostrando a la academia su invención.

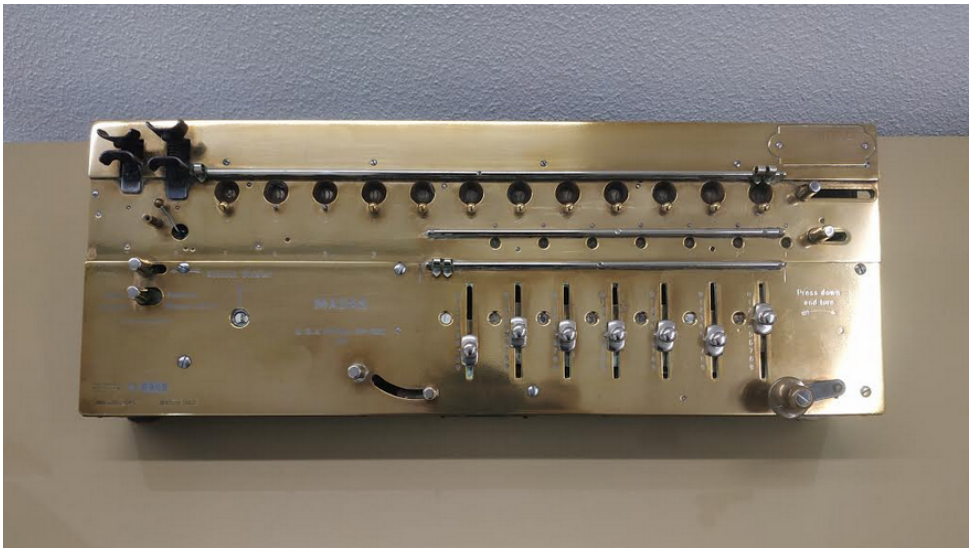

Figura 2.11: Aritmómetro expuesto en el Banco de España (Valencia). Tiene las siguientes dimensiones: 18 x 58 x 9 cm, hecho de metal y madera con un peso aproximado de 4 kilos.

Para 1842 la Máquina Analítica era tan reconocida por la academia que despertó la curiosidad de muchos eruditos. La más destacada fue Ada Lovelace condesa e hija única de Lord Byron [12]. Muchos expertos la describen como la primera programadora de la historia. Realizando varias aportaciones, en especial las publicaciones sobre la máquina analítica considerándose la programación como la más importante.

2.3.2. El aritmómetro

Charles Xavier Thomas de Colmar, destaca por patentar y construir la primera calculadora mecánica exitosa comercialmente en el mundo desde 1851 hasta 1915. Muchos personajes destacables lo trataron de hacer, tales como Pascal, Leibniz o Morland, pero ninguno consiguió el éxito comercial debido a varios defectos de producción, falta de mano de obra especializada y materiales de fabricación con alto coste. Además de un factor muy importante, la falta de necesidad para el uso de una calculadora en la sociedad «pre-industrial» convirtiéndose así en un imposible para su comercialización. Con la aparición de la Revolución Industrial a mitad del siglo XIX, surgieron nuevas necesidades tecnológicas en empresas, instituciones académicas, gubernamentales y militares generando una alta demanda de calculadoras [12].

Thomas tuvo interés en crear una calculadora mecánica versátil, por ello en 1820 presenta la patente a la "Société d'encouragement pour l'industrie nationale". Al principio se centró en el descubrimiento de multiplicar varios dígitos a la vez, consiguiéndolo más adelante con el desarrollo de varios prototipos. El inventor, al no ser experto, decidió contratar artesanos relojeros para desarrollar su idea, pero muchos le decepcionaban debido a que fabricaban mal los componentes, por ese motivo el mecanismo fallaba generando tensiones entre el desarrollador y el inventor. Durante la evolución de su segunda versión, Thomas conoce a Piolaine, un relojero experto quien se encarga de continuar con la fabricación, bajo el control del inventor completándose finalmente en 1848. El dispositivo termina-

do tuvo éxito en las numerosas pruebas que se sometía, pero aún le dificultaba la complejidad del sistema debido a que la máquina era difícil de manejar. Inmediatamente para la siguiente versión comenzaron a construir una máquina más simple. Sin embargo, Thomas se encontró con varios problemas, uno de ellos la muerte de su mejor desarrollador Piolaine en 1848, sin poder concluir con el nuevo proyecto de calculadora. A pesar de los problemas, en 1858 terminó su tercera versión con varias mejoras en los mecanismos de conteo, simplificando las operaciones de multiplicación y división. Entre 1851 y 1914 la producción en serie era continuo llegando a fabricar más de 5000 calculadoras mecánicas, vendiéndose casi la mitad de las existencias a Francia. La industria que construyó Thomas fue tan reconocida que eran los únicos proveedores de repuestos y productos finales en todo el mundo [17].

Como se puede observar, en la Figura 2.11 tenemos la parte detallada del aritmómetro, donde se divide en dos partes:

1. La parte superior contiene. 12 visualizadores que muestran el resultado de la anterior operación ocupados por todo el espacio de forma lineal; 7 visualizadores en el lado inferior derecho donde muestra el cociente final; 12 botones ubicados debajo de cada visualizador para el ajuste individual de cada cifra; 2 botones especiales en los lados extremos, el derecho usado como contador de operaciones y el izquierdo como reinicio del acumulador.

2. La parte inferior contiene. ocho palancas deslizantes con escalas numéricas del 0 al 9 utilizadas para introducir los valores; una palanca de control en el lado izquierdo para seleccionar la operación aritmética; y una manivela en el extremo derecho para ejecutar el mecanismo.

2.3.3. Calculadoras mecánicas tipo Odhner

Figura 2.12: Parte del mecanismo pin-wheel con nueve pasadores retráctiles. Los dos pasadores de la derecha son parte del mecanismo de acarreo de decenas.

Un nuevo avance en el diseño y producción de calculadoras mecánicas se dio con el perfeccionamiento del sistema de rueda de Leibnitz, convirtiéndose en un mecanismo circular, ligero y compacto.

El propósito de este diseño es tener un sistema que permita al registro de resultados girar a través de posiciones numéricas (0 a 9) dependiendo del mecanismo de configuración al que contiene un número manipulado [12]. En la Figura 2.12 tenemos una parte del mecanismo «pin-wheel», pudiéndose observar un disco circular con nueve «pines retráctiles» ubicados en la cuarta parte de la circunferencia. Estos pines se pueden retraer o extender con un mecanismo de ajuste que funciona con palancas giratorias. Los «pin-wheels» y los anillos de fijación se ensamblan en un cilindro, pudiéndose mover en distintas direcciones en función del giro de la manilla. Para avanzar un registro al número elegido, los «pines extendidos» funcionan como dientes de engranaje [18].

En el siglo XVII, el concepto de engranajes dentados lo conocía Leibnitz pero no logró desarrollar una calculadora con este sistema, hasta que a mediados del siglo XIX otros inventores desarrollaron calculadoras mecánicas con este mecanismo, pero sin producirse nunca en masa. Finalmente, a finales de siglo XIX aparecieron dos inventores importantes, el estadounidense Frank Baldwin y el sueco Willgodt Odhner[2]. Ellos desarrollaron al mismo tiempo e individualmente sus propias calculadoras, convirtiéndose la calculadora de Odhner en la más famosa comercialmente. Por lo tanto, la calculadora «pin-wheel» paso a llamarse comúnmente «calculadoras Odhner», debido al reconocimiento por ser la máquina con mejor precio y eficiente en el mercado. Durante el desarrollo de este tipo de calculadora hubo discrepancias entre ambos inventores por la patente, pero finalmente se solucionó cuando ambas máquinas fueron analizadas. En el análisis mostraron que el mecanismo de Odhner era mucho mejor que el de Baldwin [18].

Desde 1874 hasta 1892, Odhner desarrollaría calculadoras mecánicas baratas y de calidad, pero por problemas personales y empresariales decidió vender su patente a una empresa alemana. Esta empresa continuó con el desarrollo de calculadoras mecánicas, logrando el éxito con la calculadora "Brunsviga" (Figura 2.13) [18]. A partir de aquí este tipo de mecanismo perduraría hasta más de 70 años, con distintas versiones de máquinas calculadoras que muchos otros fabricantes trataron de hacer, en especial la compañía Facit (The Facit Company), de la que se hablará detalladamente en el siguiente capítulo.

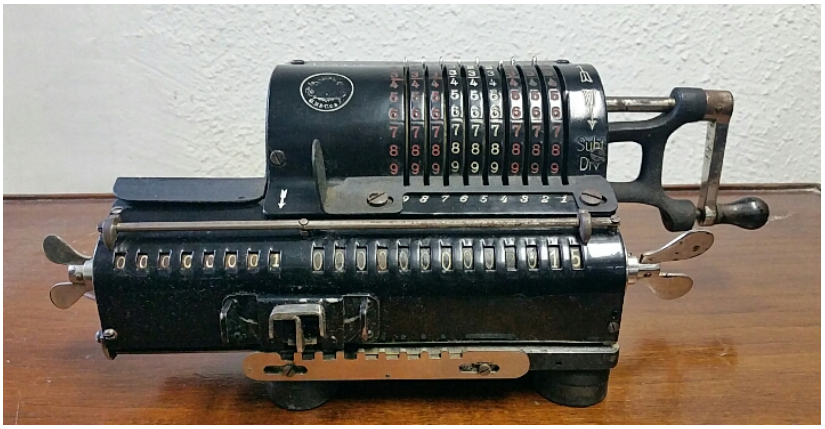

Figura 2.13: Calculadora Brunsviga Modelo A de 1898. Modelo construido a partir del diseño de Odhner (1890) distribuido por Brunsviga.

[2]El siguiente vídeo (en inglés) muestra con más detalle el mecanismo «pin-wheel»: https://www.youtube.com/watch?v=YXMuJco8onQ

CAPÍTULO 3
The Facit Company

Este capítulo está dedicado a una de las compañías más importantes en la fabricación de calculadoras mecánicas del siglo XX. Mostraremos su historia de desarrollo empresarial, la innovación de sus mecanismos aplicadas a sus distintas versiones y una descripción de los modelos más destacables, teniendo entre los ejemplos a las calculadoras mecánicas Facit C1-13 y Facit CA1-13, las cuales han sido elegidas para ser implementadas como simuladores en Scratch.

3.1 La compañía

En la primera década del siglo XX, la ciudad de Åtvidaberg (Suecia) como otras ciudades en Europa, comienza su desarrollo industrial, apareciendo múltiples compañías. Sin embargo, la aparición de la Primera Guerra Mundial y el desarrollo de la Revolución Rusa hizo que las empresas perdieran mercado, llegando muchas a la bancarrota. Una de estas compañías fue «AB ÅTVIDABERGS FÖRENADE INDUSTRIERS» creada en 1906, la cual se mantuvo con continuos problemas hasta su bancarrota en 1922, volviendo nuevamente a la actividad gracias al Banco Sydsvenska, renombrando a la organización como «AB Åtvidabergs Industrier». En la reconstrucción de la compañía, Elof Ericsson (Figura 3.1) destaca mucho en la actividad de reflote, convirtiéndose en Director General, transformando la organización en una de las más poderosas dentro de la ciudad [4].

Al mismo tiempo, una pequeña fábrica asumida por Axel Wibel y establecida en Estocolmo produce calculadoras del tipo «pin-wheels» cimentado en el mecanismo de Odhner desde alrededor de 1918 bajo el nombre de Facit AB. Durante el apogeo, Facit AB estaba a punto de entrar en bancarrota hasta que formó parte de «AB Åtvidabergs Industrier» en 1924, trasladando su producción de calculadoras mecánicas a Åtvidaberg. La organización fue tomando forma, comenzando al principio con la fabricación de distintos productos tales como: parquets, piezas de madera, cajas de radio, accesorios de taller, carrocerías de camiones y pocas calculadoras mecánicas tipo «pin-wheels». Más adelante ampliaron su producción para fabricar muebles suecos de oficina, siendo esta la que más importancia tuvo antes del desarrollo de su primero modelo de calculadora.

Las calculadoras mecánicas no fueron la principal prioridad para AB Åtvidabergs Industrier, dejando que Axel Wibel y su colaborador Karl Rudin continua-

Figura 3.1: Elof Ericsson (1887-1961), fundador y Director General de AB Åtvidabergs Industrier en 1922. Foto tomada durante una conferencia de ventas en 1927.

ran fabricando libremente. Sin embargo, el interés y el avance potencial tuvo en 1932 con el desarrollo de la calculadora de diez teclas Facit T, inventado por Karl Rudin, vendiéndose muy bien en el mercado, continuando su desarrollo hasta convertirse en las décadas siguientes en una calculadora electromecánica. Durante este tiempo, AB Åtvidabergs Industrier fue mejorando como organización, ganando posiciones en el mercado con la compra estratégica de compañías importantes tales como la fábrica de máquinas de escribir Halda-Norden, adquirida en 1938, y la compañía Original-Odhner comprada en 1942 junto con la gama de calculadoras. En consecuencia, acaparó la gama de máquinas de oficina, vendiéndose muchas con el logo de Odhner y el nombre de Facit (como por ejemplo los modelos CA1-13 y CM2-16). Cerca de 1950, la filial Facit AB paso a llamarse AB Åtvidabergs-Facit. Más tarde en 1957, después de 35 años de éxitos en la compañía, Elof Ericsson fue reemplazado por su hijo Gunnar Ericsson, dirigiéndola hasta 1970. La filial cada vez tenía más importancia con el éxito de ventas, convirtiendo al grupo en 1960 en una compañía global con decenas de sucursales en todo el mundo, y de este modo toda la organización empresarial pasó a llamarse Facit AB en 1966 [4].

Las ventas colapsaron a principios de 1970, generando una crisis en Facit AB. Entre las principales posibles causas está la compra costosa de Addo, la competencia japonesa con las calculadoras electrónicas y la negación a entender la

superioridad de los modelos electrónicos frente a los mecánicos. Como resultado, en 1973 Electrolux compra Facit AB continuando el desarrollo de máquinas electrónicas como computadoras de escritorio, medios de almacenamiento, impresoras matriciales y ordenadores personales. Finalmente fue disuelta en 1999 antes de ser comprada por Telekom Advanced Systems [4].

3.2 Evolución de los modelos mecánicos Facit

3.2.1. Primeros modelos

Karl Viktor Rudin diseñó las primeras calculadoras «Facit» ("Solución" en sueco) siendo fabricadas por la compañía de Axel Wibel en Estocolmo. En la pequeña fábrica, Rudin empezó a desarrollar su propia máquina, con el nombre de "Facit Original". Las primeras calculadoras fueron fabricadas en la filial «Facit AB» de Wibel cuando se fundó en 1918, con entregas a pedidos en 1920. Los primeros modelos fueron diseñados a partir de las patentes DE7393 y DE64925 de Willdogt T. Odhner de San Petersburgo. A pesar de esto, la calculadora tiene características y mejoras que la diferencia de otros fabricantes, siendo desarrollado mecanismos de bloqueo inverso para evitar que la manivela retroceda (DE355198), y ofreció una solución al problema del cuentarrevoluciones (DE345070). Tampoco contó con una pantalla de control para los dígitos introducidos, registrando Rudin una nueva patente (DE339121) para ser incorporada más tarde. Después de la anexión de «Facit AB» al grupo «AB Åtvidabergs Industrier» en 1924, Rudin continuó con su primera máquina, realizando mejoras en el mecanismo tal como el cambio de cociente automático (DE477002) patentada en 1929 e instalada en el Modelo S (Figura 3.3). En la Tabla 3.1 se muestra con detalle los primeros modelos Facit [19, 20].

Figura 3.2: Facit modelo Standard. Calculadora basada en el mecanismo Odhner, sin teclado. Los dígitos se introducen con pequeñas palancas que se deslizan verticalmente.

Modelo	Características	Patentes
Facit Standard Figura 3.2	Dígitos: 9 x 10 x 15 Dimensiones: 17 cm x 17 cm x 14 cm Peso: 6.8 kg Fabricación: Suecia, 1924 - 1931	DE7393 DE64925 DE355198 DE345070 DE339121
Facit S Figura 3.3	Dígitos: 10 x 8 x 13 Dimensiones: 17.5 cm x 13 cm x 12 cm Peso: 5.3 kg Fabricación: Suecia, 1935 - 1954	DE7393 DE64925 DE355198 DE345070 DE339121 DE477002

Tabla 3.1: Primeros modelos Facit.

3.2.2. Modelos de 10 teclas

Para introducir números en los primeros modelos Facit, era necesario usar pequeñas palancas conectadas cada una a su rueda dentada, siendo problemático y poco práctico. Por esta razón, durante mucho tiempo Rudin buscaba una manera de conectar las ruedas dentadas con un teclado mecánico, consiguiendo una solución al "problema de 10 teclas" patentándola en 1929 (patente DE535576).

En el Modelo T (Figura 3.5) que salió al mercado en 1932 se utilizó este mecanismo de ajuste (llamado «Principio Facit»). La patente DE535576 del Modelo Facit T ("T" sueco "Tangent", "Tecla") tiene tres innovaciones importantes al modelo de Odhner:

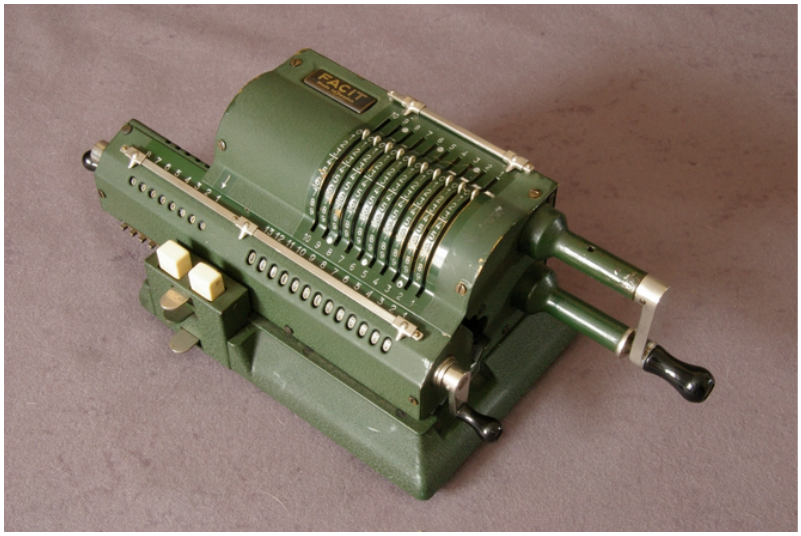

Figura 3.3: Facit modelo S. Parecido al modelo 7 de Odhner. Tiene un mecanismo extra de limpieza para el rotor en forma de tecla, ubicado en la parte inferior izquierda.

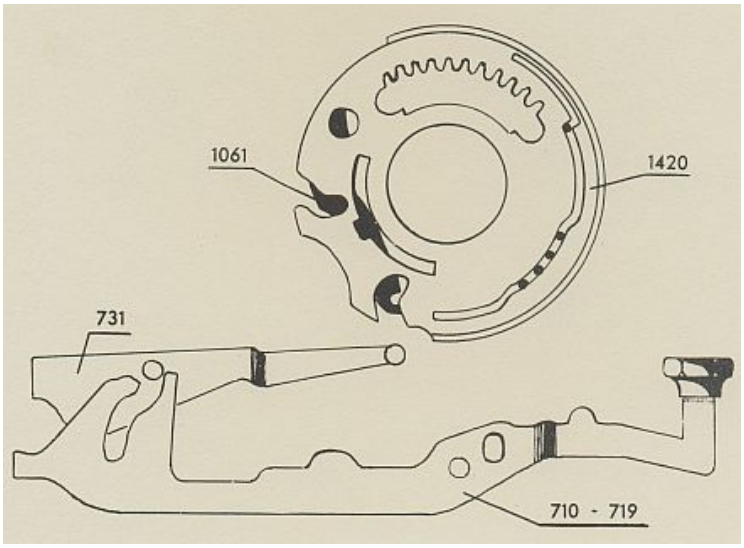

Figura 3.4: Rueda dentada del Modelo T. Llamado también «Principio Facit», boceto sacado de la patente DE535576.

Primero: El mecanismo de ajuste (rotor de la rueda dentada) se monta de forma deslizante sobre una corredera en dirección axial, frente a los dos contadores. La corredera de la ranura se desliza sobre un rodamiento de bola, moviéndose de derecha a izquierda cuando se teclea.

Segundo: Una rueda dentada puede girar en dos direcciones, una derecha y otra izquierda, provocando en función de la dirección una configuración de valores diferentes. Por eso se usa la "rueda dentada dividida" reduciendo a la mitad el ángulo de rotación de la rueda, agregando a cada rueda nueve dientes ajustables, donde solo cuatro funcionan de forma habitual (se mueven a lo largo de los canales radiales en el cuerpo de la rueda dentada). En cambio, los cinco dientes restantes de cada rueda están conectados firmemente. Cuando se ingresan los números del 1 al 4, la rueda dentada gira a la izquierda seleccionando el sector del diente de la rueda el número introducido por la tecla; en cambio, al ingresar los números del 5 al 9, la rueda dentada gira en sentido contrario seleccionando nuevamente el sector del diente de la rueda del número tecleado.

Tercero (ver Figura 3.4): El mecanismo de ajuste al presionar una tecla, el brazo pequeño (731) pivota gracias al movimiento del segundo brazo (710-719) cuya parte trasera tiene una ranura de leva. La parte derecha del brazo pequeño (731) se engancha con la ranura del disco (1420), girando el disco dentado hasta el dígito tecleado. Cada brazo (710-719) tiene una ranura de forma y longitud diferente que conecta con la leva del brazo (731): cuanto mayor tamaño tenga la ranura el número será de una cantidad mayor. De este modo se determina el ángulo de rotación del brazo (731). Si tecleamos los números del 1 al 4, el brazo de ajuste izquierdo funciona hacia abajo. Por el contrario, al seleccionar del 5 al 9, el brazo de ajuste derecho gira hacia arriba. Si se termina de presionar una tecla, el carro (1061) ubicado dentro de la rueda dentada se mueve hacia la izquierda, desplazando la leva del brazo (731). Cuando la tecla 0 es presionada, el mecanismo de ajuste no gira el disco dentado (1420).

Figura 3.5: Diseño de los modelos T y TK. La "K" proviene del sueco "Kvotkoppling" que significa "cambio de cociente deslizante". El modelo lleva instalado el teclado Dalton.

Figura 3.6: Diseño de los modelos NTK y C1-13. El modelo C1-13 es internacionalmente conocido con más de 300.000 unidades vendidas.

Con este mecanismo de ajuste es necesario una configuración para el teclado numérico, la primera fila con los números 2 – 4 – 5 – 7 – 9 y la segunda fila con los números 1 – 3 – 0 – 6 – 8. De esta manera los números del 1 al 4 están conectados a los brazos de ajuste izquierdo y el conjunto de números del 5 al 9 con el derecho. En el lado izquierdo del teclado hay dos teclas de tabulación (dos flechas) para poder configurar las operaciones de suma, resta y multiplicación; a los lados de la calculadora están las palancas para poner los registros a cero (patente DE575837), y en 1942 se agrega el cambio de cociente deslizante (patente DE719178). En esta versión se desarrollan los contadores con capacidad de 9 dígitos para el mecanismo del registro de entrada, 8 dígitos para el mecanismo del registro cociente, y 13 dígitos para el mecanismo del registro acumulador (modelo 9 x 8 x 13).

La innovación de Rudin con la calculadora de diez teclas le dio ventaja frente a otras calculadoras mecánicas, es decir, ingresar los números con un teclado hizo que fuera práctico (mano izquierda teclea – mano derecha escribe). Además generó mayor seguridad en la configuración mecánica de las operaciones aritméticas: el sistema compacto de la mecánica hizo que fuera una calculadora muy portable y su carcasa cerrada a prueba de polvo y daños aportó mayor resistencia. Con este diseño comenzó el punto de partida para una generación de calculadoras Facit tal y como se muestra en la Tabla 3.2.

Figura 3.7: Diseño de los modelos LX, NLX y C1-19. Su mecanismo le permite almacenar los resultados a lo largo del acumulador de 19 dígitos.

Figura 3.8: Diseño del modelo EK. Primer modelo con motor eléctrico de 400 revoluciones por minuto.

En 1938, durante la evolución del modelo Facit TK, se desarrollaron varios modelos donde el rango de dígitos fue aumentado (19 dígitos) para los contado-

Modelo	Características	Patentes
Facit TK Figura 3.5	Dígitos: 9 x 8 x 13 Dimensiones: 21 cm x 18.5 cm x 14.5 cm Peso: 6.0 kg Fabricación: Atvidaberg -Facit, Suecia 1936 - 1954	DE535576 DE719178 DE575873
Facit NTK Figura 3.6	Dígitos: 9 x 8 x 13 Dimensiones: 22 cm x 20 cm x 15 cm Peso: 6.3 kg Fabricación: Atvidaberg -Facit, Suecia 1954 - 1957	DE535576 DE719178 DE575873
Facit C1-13 Figura 3.6	Dígitos: 9 x 8 x 13 Dimensiones: 22.5 cm x 21 cm x 15 cm Peso: 6.6 kg Fabricación: Atvidaberg -Facit, Suecia 1957 - 1972	DE535576 DE719178 DE575873

Tabla 3.2: Modelo T de 10 teclas.

res de registro, pudiendo hacer operaciones con números mucho más grandes, dando como resultado el modelo Facit LX (Figura 3.7). El mecanismo patentado por Bengt Carlstrom (DE703464 y DE735954) hizo que este tipo de calculadoras fuera interesante para científicos o ingenieros, ya que permite operaciones con números con muchos dígitos. Después de algunos cambios en el diseño, esta calculadora fue vendida con el nombre de NLX y mas tarde como el modelo C1-19. Mírese la Tabla 3.3.

3.2.3. Motorización de los modelos

Los modelos mecánicos de 10 teclas continuaron su evolución y éxito de ventas. Sin embargo, para principios de 1930 decidieron ir mas allá, innovando un modelo experimental automático llamado Facit EK (Figura 3.8) fabricada en 1936. Esta versión electromecánica del Modelo T es muy parecida a la versión mecánica, a diferencia que el tambor de la rueda dentada gira con el movimiento de un motor eléctrico, eliminando la manivela para operar. La instalación de este sistema resultó fácil para los desarrolladores; no obstante, la dificultad aparece en el control del tambor de la rueda dentada. Esta rueda, como se detalló anteriormente tiene que rotar en direcciones diferentes para procesar las operaciones, y en principio el motor solo giraba en una única dirección.

Figura 3.9: Diseño de los modelos NEA y NE. Esta versión realiza las divisiones de forma automática, y las multiplicaciones en semiautomática.

El problema se solucionó con el desarrollo de un mecanismo de acoplamiento que Rolf Erik y Bengt Carlstrom patentaron (DE656737 y DE682246). Este mecanismo tiene dos ruedas dentadas conectadas al motor montadas sobre un eje, y en la interfaz de usuario existen dos teclas de control (+ y -). Cuando se presiona alguna de las teclas, acciona una varilla de empuje conectando un trinquete en el eje de transmisión de la rueda dentada que le corresponde, girando arriba o abajo en función de la tecla seleccionada. El modelo E de Facit, aparte de tener la tecla de función + (usado para multiplicar) y la tecla – (usado para dividir), lleva instalado una palanca en la parte izquierda que ajusta la operación aritmética con la posición inferior para suma/resta y la posición superior para multiplicación/división. Finalmente, para limpiar los registros de los contadores hay una tecla ubicada en el extremo derecho de la calculadora, eliminando las palancas de reinicio del lado derecho e izquierdo. A partir de este diseño la gama de calculadoras con motor eléctrico fue evolucionando.

Las calculadoras Facit con motor eléctrico no eran totalmente automáticas porque las operaciones de multiplicación o división se realizaban de forma semiautomática. Por eso en 1945 aparece la primera calculadora mecánica con diez teclas, llamada Facit ESA (Elektriska Super Automaten) (Figura 3.10). Este modelo revolucionó el uso de la calculadora moderna, teniendo similitud con el proceso actual de realizar operaciones aritméticas de una calculadora de escritorio actual, es decir, cuando se realiza una multiplicación se introduce el primer valor seguido del símbolo “x” y después el segundo valor para finalmente teclear el símbolo “=”, obteniendo el resultado de la operación.

Los diseñadores Erik Konrad y Sture Efraim Toorell buscaban una manera de convertir el modelo Facit NEA en una calculadora automática. Por esto, en 1947 patentan el mecanismo de multiplicación y división automáticas (DE922553). A la hora de fabricar el modelo ESA se dieron cuenta del ahorro de espacio que libera-

Modelo	Características	Patentes
Facit LX Figura 3.7	Dígitos: 10 x 10 x 19 Dimensiones: 22 cm x 21 cm x 16 cm Peso: 7.8 kg Fabricación: Suecia, 1938 - 1954	DE703464 DE726893 DE735954
Facit NLX Figura 3.7	Dígitos: 10 x 10 x 19 Dimensiones: 22 cm x 21 cm x 16 cm Peso: 7.9 kg Fabricación: Suecia, 1954 - 1956	DE703464 DE726893 DE735954
Facit C1-19 Figura 3.7	Dígitos: 10 x 10 x 19 Dimensiones: 22 cm x 21 cm x 16 cm Peso: 7.9 kg Fabricación: Suecia, 1956 - 1960	DE703464 DE726893 DE735954

Tabla 3.3: Modelos de 10 teclas con carro de alta capacidad.

ba el mecanismo semiautomático multiplicador. Además, añadieron una mejora en la operación multiplicadora, donde un número puede multiplicarse con varios factores, almacenando el dato en el mecanismo multiplicador, liberándose al presionar la tecla SUB-STOP. En 1949 el modelo Facit ESA-0 aparece con varias mejoras: el limpiador de los contadores a cero se realiza presionando las teclas I, II, III, eliminando las dos palancas ubicadas en el lado lateral izquierdo y la tecla "0" del lado derecho de la calculadora; se agrega una palanca de control con las posiciones MULT-ADD-DIV y otra secundaria que cumple la misma función que el modelo Facit NEA.

3.2.4. Modelos de la serie C2

Todos los modelos Facit tenían el mismo mecanismo de diez teclas hasta mitad de 1950. Después la compañía tuvo que dar respuesta a las amenazas competitivas cambiando el mecanismo. Los factores de cambio fueron:

1. La expiración de la patente del modelo T, dejando que otros fabricantes puedan usar el "principio Facit" en el diseño de sus propias calculadoras con teclado.

Modelo	Características	Patentes
Facit EK Figura 3.8	Dígitos: 9 x 8 x 13 Dimensiones: 26.5 cm x 25 cm x 16.5 cm Peso: 9.6 kg Fabricación: Atvidaberg -Facit, Suecia 1936 - 1943	DE535576 DE656737 DE719178 DE682246
Facit NEA Figura 3.9	Dígitos: 9 x 8 x 13 Dimensiones: 26.5 cm x 25 cm x 16.5 cm Peso: 10.2 kg Fabricación: Atvidaberg -Facit, Suecia 1953 - 1956	DE535576 DE656737 DE719178 DE682246
Facit NE Figura 3.9	Dígitos: 9 x 8 x 13 Dimensiones: 26.5 cm x 25 cm x 16.5 cm Peso: 9.6 kg Fabricación: Atvidaberg -Facit, Suecia 1953 - 1956	DE535576 DE656737 DE719178 DE682246

Tabla 3.4: Modelos E de 10 teclas motorizados.

2. Competencia al modelo T, bajando la demanda de calculadoras de rueda dentada con teclado; entre los modelos competidores tenemos a Everest Z4 y Precisa 117.

3. Competencia al modelo automático CA1-13, donde las compañías SCM y Olympia desarrollaron máquinas totalmente mecánicas con teclado a precios competitivos.

4. Incipiente aparición de la informática electrónica, con modelos compactos de calculadoras.

Como respuesta a las amenazas, la compañía desarrolla los modelos descritos en la Tabla 3.12 y Tabla 3.13. El mecanismo patentado (DE1082754 o De1051539) por Erik Konrad Grip hizo que el modelo CM2-16 fuera la primera calculadora de manivela con doble transmisión, es decir, el valor del resultado puede ser procesado de inmediato, como en una calculadora electrónica.

Figura 3.10: Diseño del modelo ESA-0. Esta versión tiene aplicada la patente DE922553 para la realización de multiplicaciones automáticas.

Figura 3.11: Diseño del modelo CA1-13. La caja metálica es remplazada por una cubierta fundida a presión. Mantiene la misma mecánica que el modelo ESA-0

Figura 3.12: Diseño de los modelos CM2-16, CM2-13S y 10-04.

Modelo	Características	Patentes
Facit ESA-0 Figura 3.10	Dígitos: 9 x 8 x 13 Dimensiones: 28 cm x 26 cm x 18 cm Peso: 12.4 kg Fabricación: Suecia, 1949 - 1956	DE535576 DE656737 DE871079 DE922553
Facit CA1-13 Figura 3.11	Dígitos: 9 x 8 x 13 Dimensiones: 29 cm x 28.5 cm x 19.5 cm Peso: 12.7 kg Fabricación: Suecia, 1956 - 1973	DE535576 DE656737 DE871079 DE922553

Tabla 3.5: Modelos motorizados automáticos.

Modelo	Características	Patentes
Facit CM2-16 Figura 3.12	Dígitos: 11 x 9 x 16 Dimensiones: 24 cm x 27.5 cm x 15.5 cm Peso: 7.8 kg Fabricación: Suecia, 1959 - 1967	DE1051539 DE1082754
Facit CM2-13S Figura 3.12	Dígitos: 9 x 8 x 13 Dimensiones: 24 cm x 27.5 cm x 15.5 cm Peso: 7.5 kg Fabricación: Suecia, 1964 - 1965	DE1051539 DE1082754
Facit 10-04 Figura 3.12	Dígitos: 11 x 9 x 16 Dimensiones: 24 cm x 27.5 cm x 15.5 cm Peso: 7.5 kg Fabricación: Suecia, 1967 - 1969	DE1051539 DE1082754

Tabla 3.6: Modelos serie C2 manuales.

Figura 3.13: Diseño de los modelos CA2-16, CA2-16SX y 10-07. Modelo que funciona con un motor de 420 revoluciones. Contiene un dispositivo de memoria mecánico.

Modelo	Características	Patentes
Facit CA2-16 Figura 3.13	Dígitos: 11 x 9 x 16 Dimensiones: 27.5 cm x 35 cm x 19 cm Peso: 15.5 kg Fabricación: Suecia, 1962 - 1969	DE1051539 DE1082754 DE871079 DE922553
Facit CA2-16SX Figura 3.13	Dígitos: 11 x 9 x 16 Dimensiones: 27.5 cm x 35 cm x 19 cm Peso: 15.4 kg Fabricación: Suecia, 1965 - 1967	DE1051539 DE1082754 DE871079 DE922553
Facit 10-07 Figura 3.13	Dígitos: 11 x 9 x 16 Dimensiones: 27.5 cm x 35 cm x 19 cm Peso: 15.0 kg Fabricación: Suecia, 1967 - 1972	DE1051539 DE1082754 DE871079 DE922553

Tabla 3.7: Modelos serie C2 motorizados.

CAPÍTULO 4
Manejo de las calculadoras Facit

Para poder entender cómo se usa una calculadora mecánica Facit es necesario aprender la funcionalidad de cada uno de sus componentes. Por eso en este capítulo se va a exponer cómo realizar operaciones aritméticas con las calculadoras C1-13 y CA1-13. Como se detalló anteriormente, el modelo C1-13 es de tipo manual, es decir, se necesita aplicar movimientos de manivela por parte del usuario para ejecutar operaciones. Además, esta calculadora es de las más intuitivas y rápidas de aprender, abarcando poca versatilidad a la hora de ejecutar operaciones de cálculo complejas con múltiples dígitos, rangos numéricos extensos y resta con números negativos, aclarando al lector que el aprendizaje de este modelo sirve también para entender el funcionamiento de la calculadora CA1-13, siendo explicado posteriormente.

Antes de comenzar es importante comprender que este capítulo es un resumen del manejo de las calculadoras. Si el lector esta atraído en el funcionamiento detallado de los modelos CA1-13[1] y C1-13[2], los manuales de referencia oficial Facit de cada calculadora muestran de forma minuciosa el manejo con ejemplos prácticos, que pueden ser aplicados a los dos simuladores implementados de este trabajo.

Los manuales oficiales Facit destacan ciertas normas de uso antes de ejecutar una operación aritmética de forma rápida y acertada:

- Primeramente, aplicar la limpieza de dígitos en todos los registros de la calculadora (poner a 0 los tres registros con las opciones de borrado).
- Realizar los pasos de forma ordenada y precisa para poder obtener el resultado correcto de la operación (evitar saltar pasos intermedios).
- Tener cuidado con el acarreo de números al utilizar los tabuladores (desplazamiento equivocado del número), ya que puede realizar la operación de manera errónea.
- Prestar atención a la capacidad del resultado al realizar una operación, ya que los registros tiene un máximo de dígitos (modelo de registros 9 x 8 x 13) para evitar desbordamiento de cantidades.

[1] Manual disponible en alemán: https://www.manualscat.com/es/facit-ca1-13-manual

[2] Manual disponible en inglés: https://mechanicalcalculators.files.wordpress.com/2016/03/facit-c1-13.pdf

4.1 Calculadora C1-13

4.1.1. Componentes principales

El modelo C1-13 está compuesto de siete elementos, distribuidos por toda la calculadora, tal y como se muestra en la Figura 4.1:

1. Palanca de limpieza del registro producto.
2. Cambio de carro izquierdo y derecho. Mueve el número una posición en el mecanismo del registro de configuración hacia la dirección tecleada.
3. Teclas numéricas.
4. Tabulador. Mueve el número a la cabecera del registro de configuración, posición de división.
5. Palanca de limpieza del registro de configuración.
6. Palanca de limpieza del registro multiplicador.
7. Manivela de operación. Tiene dos opciones de giro, horario (suma y multiplicación) y anti-horario (resta y division).

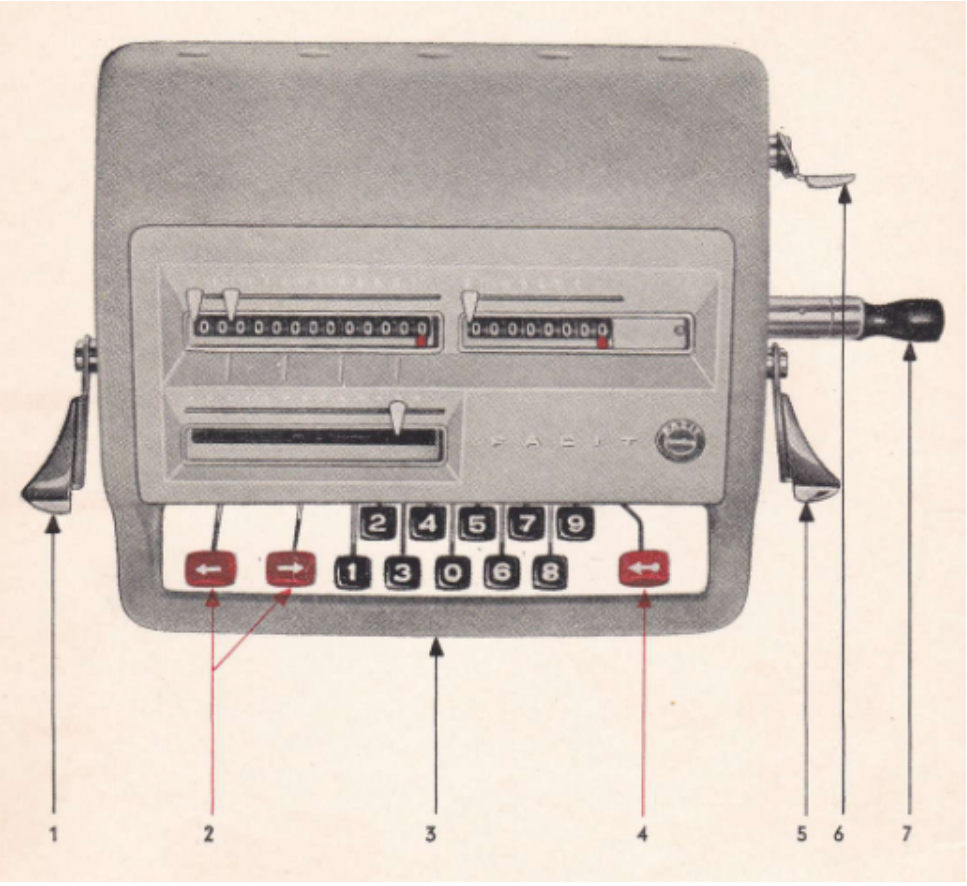

Figura 4.1: Partes principales del modelo C1-13. Extraído del manual C1-13 de 1956.

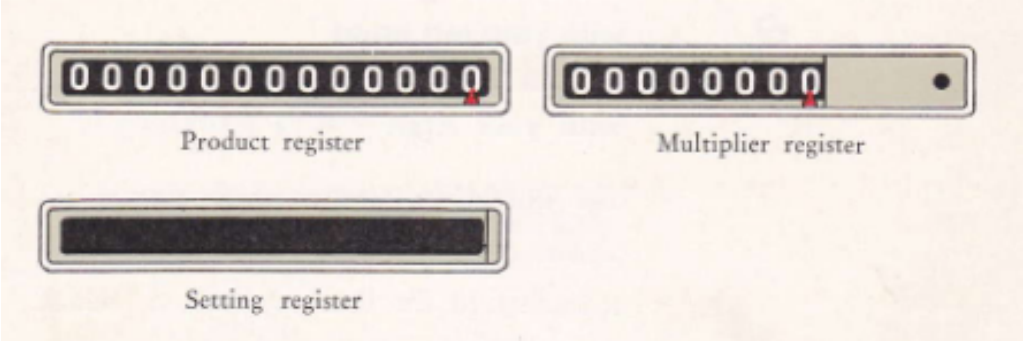

Figura 4.2: Registros principales del modelo C1-13. Extraído del manual C1-13 de 1957.

Continuando con sus componentes, este modelo tiene tres registros principales ubicados en la parte frontal superior del teclado. Mírese la figura 4.2:

Registro Producto: *Product register* muestra la respuesta cuando se suma, resta o multiplica alguna cantidad. También muestra el resto cuando se divide.

Registro Multiplicador: *Multiplier register* muestra el cociente de la división, el número de ítems introducidos en la suma y en el multiplicador de la multiplicación.

Registro de Configuración: *Setting register* graba inmediatamente cada tecleo realizado con el conjunto de teclas.

El sistema de borrado para el modelo C1-13 es principalmente con el accionamiento de palancas. Cada uno de los tres registros tiene su propia palanca de borrado. En la Figura 4.3 se muestra cómo se realiza el borrado de los registros.

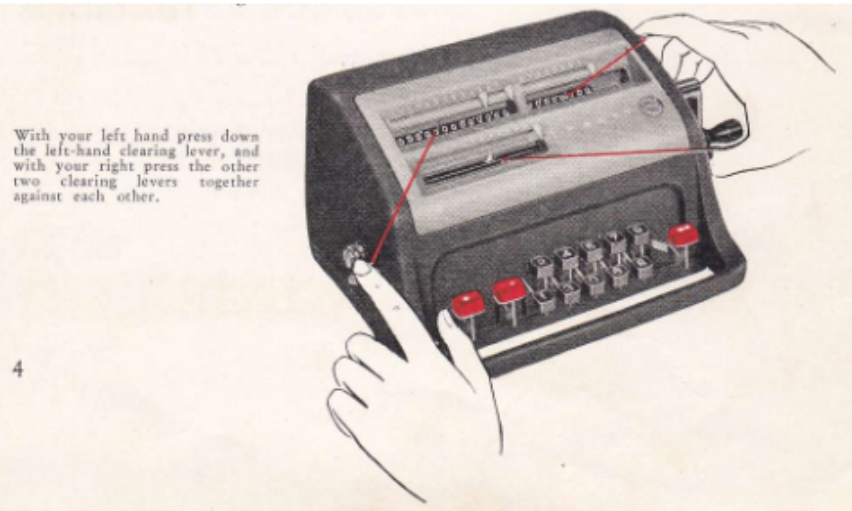

Figura 4.3: Palancas de borrado principales del modelo C1-13, utilizadas para pone a cero los tres registros señalados en el manual. Extraído del manual C1-13 de 1957.

Los números del registro de configuración introducidos pueden ser movidos paso a paso hacia la izquierda, derecha o totalmente a la cabecera del registro, mediante las teclas rojas de desplazamiento de carro y tabulador. En la Figura 4.4 se muestran las teclas con la dirección en la que se realiza el movimiento.

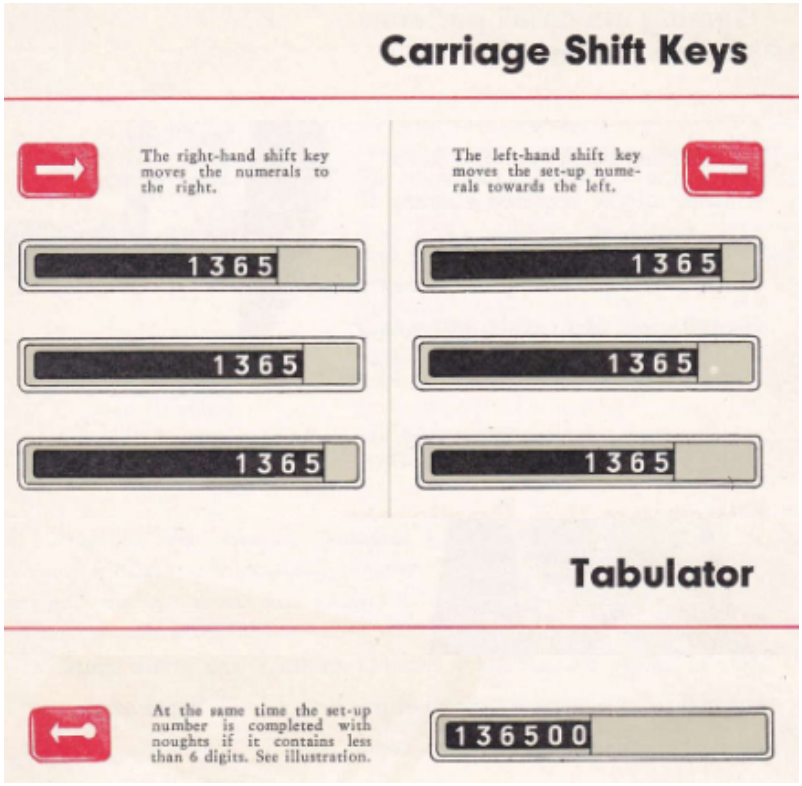

Figura 4.4: Cambio de carro derecho, izquierdo y tabulador total izquierdo del modelo C1-13. Extraído del manual C1-13 de 1957.

Para ejecutar las operaciones en la calculadora es necesario dar movimientos de manivela, ya que el mecanismo necesita transformar la configuración aplicada en un resultado. Por ello existen dos tipos de movimientos: el giro horario (suma, multiplicación) y giro antihorario (resta, división). En la Figura 4.5 tenemos la forma en como se tiene que realizar el giro.

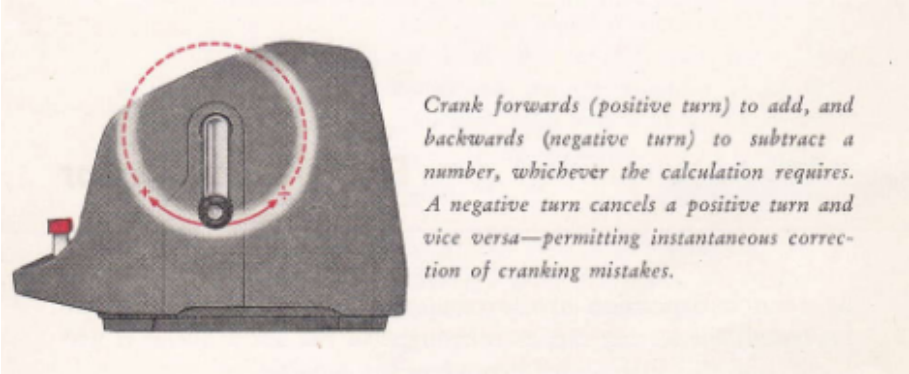

Figura 4.5: Tipos de giros para el modelo C1-13. Extraído del manual C1-13 de 1957.

4.1.2. Cálculos elementales

Para realizar una suma con dos o más cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.6:

1. Limpiamos los registros con las tres palancas de borrado.
2. Tecleamos la primera cantidad para después dar una vuelta horaria a la manivela de operación. Observamos el número agregado en el registro producto.
3. Limpiamos el registro de configuración e introducimos la siguiente cantidad con un giro horario de manivela.
4. Repetimos el paso 3 para seguir sumando cantidades.
5. Observamos el número de elementos introducidos en el registro multiplicador.

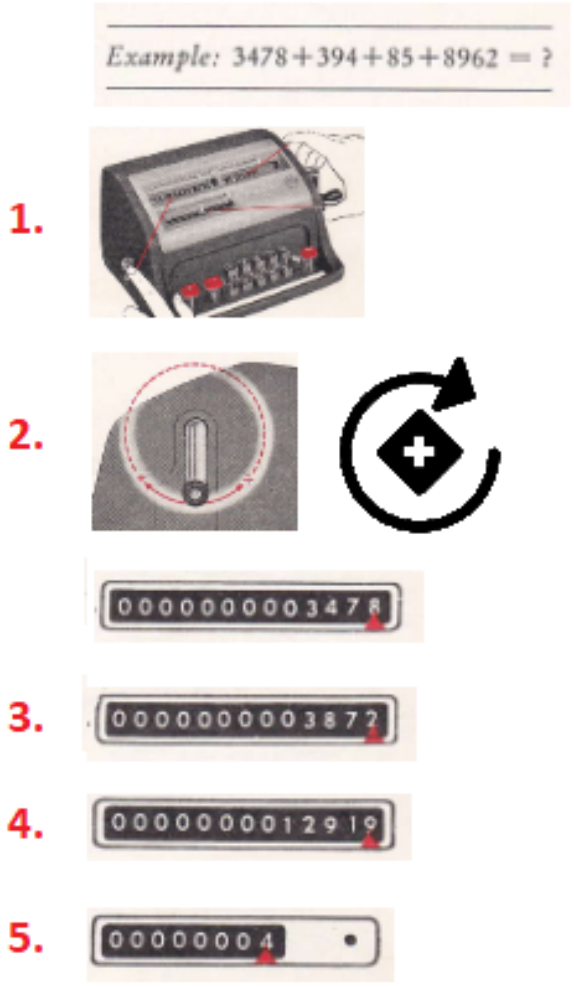

Figura 4.6: Sumar cantidades en el modelo C1-13. Extraído del manual C1-13 de 1957.

Para realizar una resta de cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.7:

1. Limpiamos los registros con las tres palancas de borrado.
2. Tecleamos la primera cantidad para después dar una vuelta horaria a la manivela de operación. Observamos el número agregado en el registro producto.
3. Limpiamos el registro de configuración para después teclear la segunda cantidad y restarla con un giro anti-horario de manivela. Observamos el resultado en el registro producto.

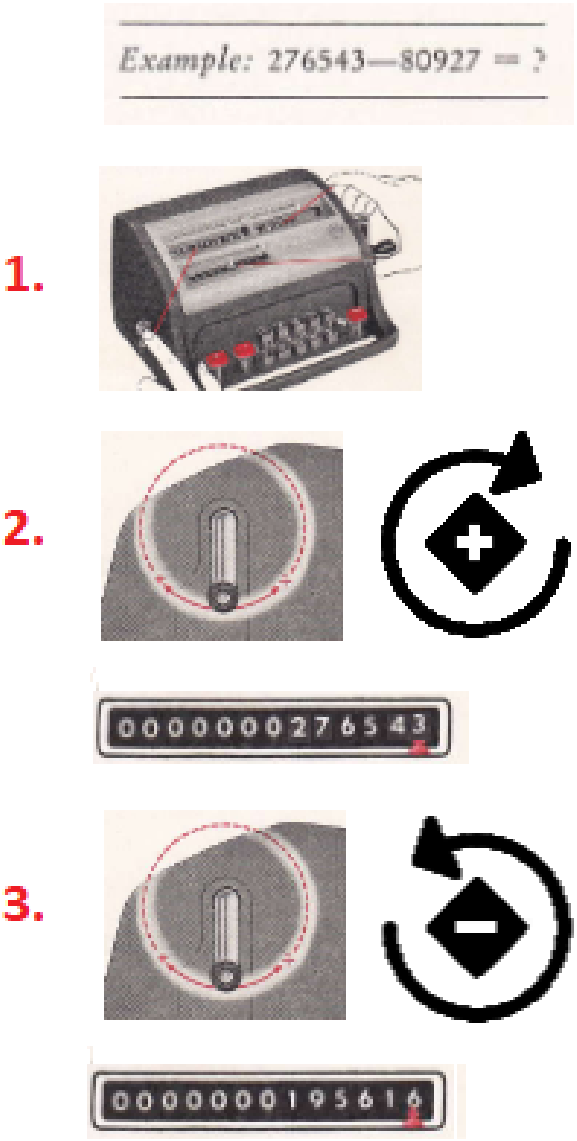

Figura 4.7: Restar cantidades en el modelo C1-13. Extraído del manual C1-13 de 1957.

Para realizar una multiplicación de cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.8:

1. Limpiamos los registros con las tres palancas de borrado.
2. Tecleamos el multiplicando. Después agregamos las unidades del multiplicador al registro multiplicador con giros horarios (este caso son 3 giros). Observamos el registro multiplicador con la unidad y el registro producto con el resultado de 6943259 x 3.
3. Tecleamos una vez el cambio de carro izquierdo y giramos 4 veces para agregar un cuatro a las decenas en el registro multiplicador.
4. Tecleamos dos veces el cambio de carro izquierdo y giramos 2 veces para agregar un 2 a los millares en el registro multiplicador. Finalmente, observamos el resultado en el registro producto.

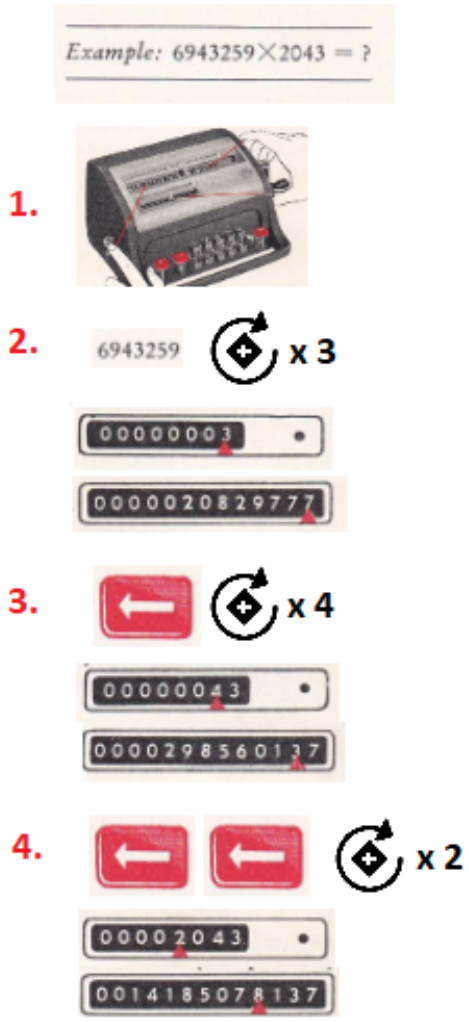

Figura 4.8: Pasos para multiplicar cantidades en el modelo C1-13. El número de giros es el valor que tiene ese dígito que se introduce. Extraído del manual C1-13 de 1957.

Para realizar una división de cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.9:

1. Limpiamos los registros con las tres palancas de borrado. Introducimos el dividendo, presionamos el tabulador y transferimos la cantidad con una vuelta horaria de manivela. Después limpiamos el registro multiplicador y configurador. Añadimos el divisor y presionamos el tabulador.
2. Realizamos giros antihorarios de manivela restando el divisor al dividendo hasta que no se pueda restar más a este último.
3. Presionamos la tecla de cambio de carro derecho una vez y continuamos con los giros antihorarios hasta no poder restar más al dividendo.
4. Volvemos a repetir el paso 3. Observamos cómo va introduciendo el resultado en el registro multiplicador.
5. Repetimos nuevamente el paso 3. Cuando el registro producto no se pueda restar más o el cambio de carro derecho no se pueda mover, paramos la operación.
6. Observamos el resultado en el registro multiplicador.

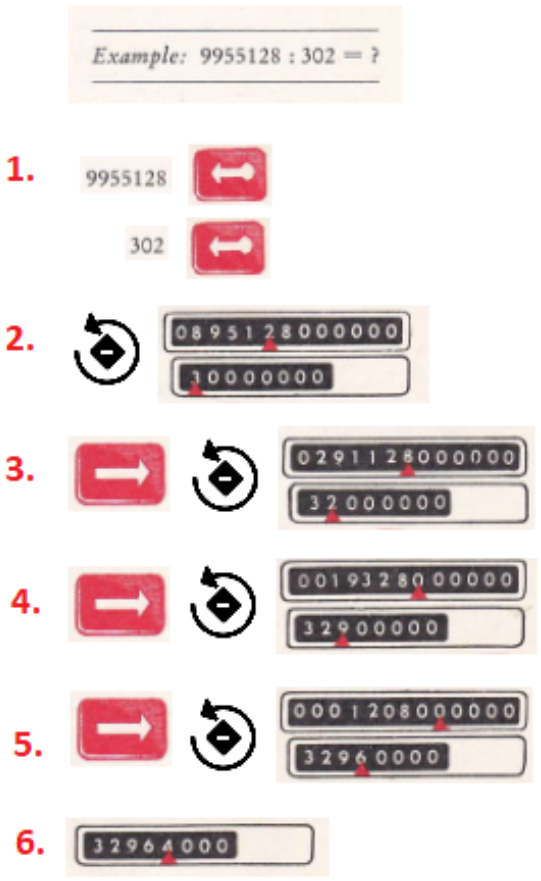

Figura 4.9: Dividir cantidades en el modelo C1-13. Extraído del manual C1-13 de 1957.

4.2 Calculadora CA1-13

4.2.1. Componentes principales

El modelo CA1-13 está compuesto de diecisiete elementos, distribuidos por toda la parte frontal de la calculadora, tal y como se muestra en la Figura 4.10:

1. Teclas de limpieza para los tres registros.
2. Palanca de control deslizante con tres posiciones. Posición izquierda para sumar, restar y multiplicar. Posición central para operaciones negativas. Posición derecha para dividir.
3. Palanca deslizante secundaria. Funciona cuando la Palanca de control está en la posición central. Cuando está en la posición izquierda no sigue el cambio de paso; en cambio la posición derecha realiza el paso de izquierda a derecha.
4. Teclas de operación 1:

   Tecla ADD : Agrega una cantidad tecleada al registro acumulador (III). El registro entrada (I) se limpia después de ser tecleada.

   Tecla ÷ : Realiza operaciones de resta y división.

   Tecla + : Realiza operaciones de suma y multiplicación (semiautomático).
5. Teclas de operación 2 (Multiplicación automática):

   Tecla X : Realiza operaciones de multiplicación.

   Tecla = : Se utiliza en la multiplicación después de introducir la segunda cantidad.
6. Desplazamiento de carro izquierdo. Mueve el número una posición en el mecanismo del registro entrada (I) hacia la izquierda.
7. Desplazamiento de carro derecho. Mueve el número una posición en el mecanismo del registro entrada (I) hacia la derecha.
8. Tabulador. Mueve el número a la cabecera del registro entrada (I), posición de división.
9. Tecla de dirección del giro del motor. Tiene dos opciones de giro, positivo (suma, resta y multiplicación) y negativo (divisiones).
10. Señal de la dirección de rotación. Color negro (giro positivo), color rojo (giro negativo).
11. Comas deslizables.
12. Puntero de ubicación del registro (puntero rojo).

13. Botón de parada (Sub-Stop). Para eliminar la configuración del registro entrada (I) al restar (se presiona simultáneamente con la tecla ÷). Y al dividir, para interrumpir la división.
14. Teclas numéricas
15. Registro Producto (III). Capacidad de 13 dígitos.
16. Registro Multiplicador (II). Capacidad de 8 dígitos
17. Registro de Configuración (I). Capacidad de 9 dígitos

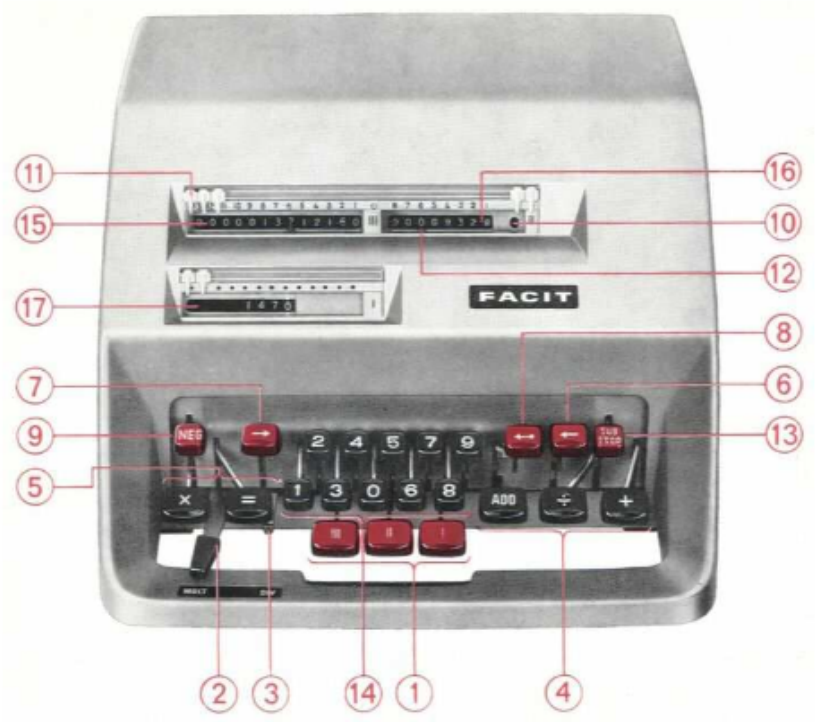

Figura 4.10: Parte principales del modelo CA1-13. Extraído del manual CA1-13 de 1956.

4.2.2. Cálculos elementales

Para poder sumar dos cantidades o más, seguiremos los siguientes pasos tal como se muestra en la Figura 4.11:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades acompañada de la tecla «ADD».
4. Observamos el resultado en el registro producto (III).
5. Contemplamos el número de elementos agregados.

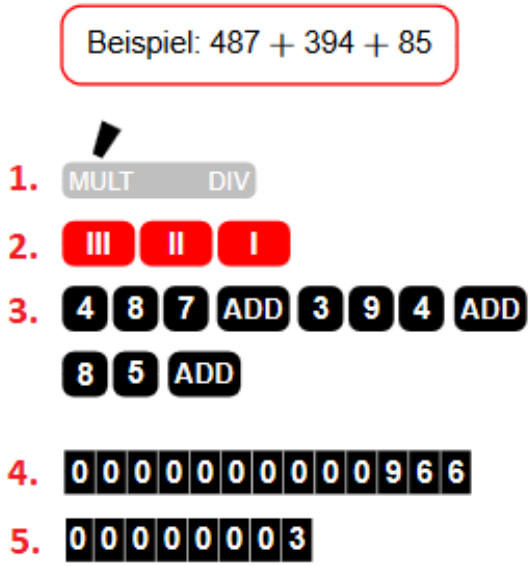

Figura 4.11: Pasos para sumar cantidades en CA1-13. La tecla «ADD» introduce de manera rápida las cantidades en el registro. Extraído del manual CA1-13 de 1956.

Para poder restar dos cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.12:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades acompañada de la tecla «ADD». Después de agregar la última cantidad, tecleamos «SUB-STOP» seguido de la tecla «÷».
4. Observamos el resultado en el registro producto (III).

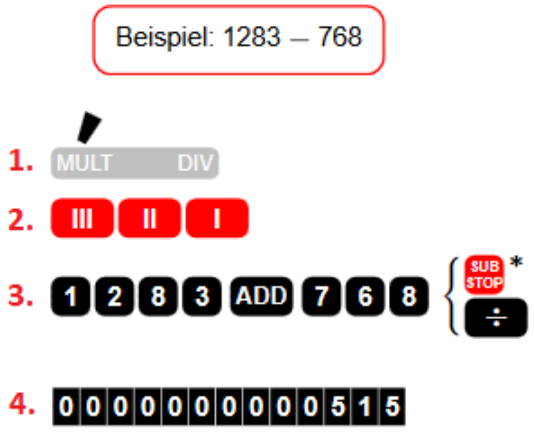

Figura 4.12: Pasos para restar cantidades en CA1-13. La tecla «SUB-STOP» se aplica para evitar realizar un cambio de carro izquierdo. Extraído del manual CA1-13 de 1956.

Para poder multiplicar dos cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.13:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades acompañada de la tecla «X». Para mostrar el resultado tecleamos la tecla igual.
4. Observamos el resultado en el registro producto (III).
5. Contemplamos el multiplicando en el registro multiplicador (II) y el multiplicador en el registro de configuración (I).

Figura 4.13: Pasos para multiplicar cantidades en CA1-13. La multiplicación para este modelo es muy similar a las calculadoras actuales. Extraído del manual CA1-13 de 1956.

Para poder dividir dos cantidades, seguiremos los siguientes pasos tal como se muestra en la Figura 4.14:

1. Colocamos la palanca de control deslizante en «DIV».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades, acompañada del tabulador total y la tecla «ADD». Para mostrar el resultado tecleamos tabulador total y «÷»
4. Observamos el resultado en el registro multiplicador (II).
5. Contemplamos el resto en el registro producto (III).

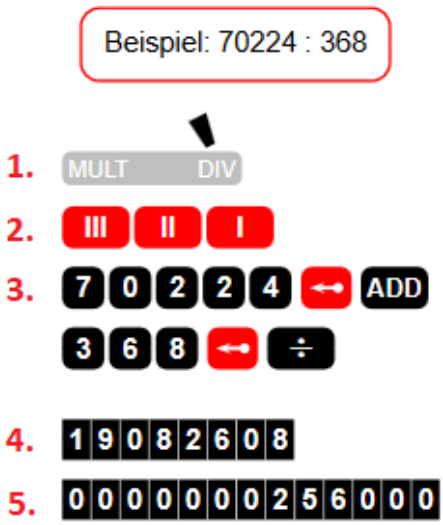

Figura 4.14: Pasos para dividir cantidades en CA1-13. Este modelo solo permite realizar divisiones de forma automática. Extraído del manual CA1-13 de 1956.

4.2.3. Cálculos especiales

Para realizar un cuadrado, seguiremos los siguientes pasos tal como se muestra en la Figura 4.15:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos la cantidad, acompañada de la tecla igual.
4. Observamos el resultado en el registro producto (III).

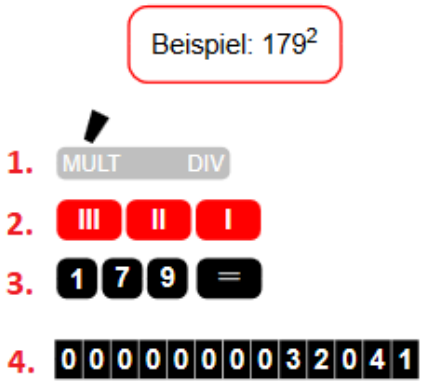

Figura 4.15: Pasos para realizar solo funciónes cuadradas en CA1-13. Extraído del manual CA1-13 de 1956.

Para sumar y restar cantidades de diez a trece dígitos, seguiremos los siguientes pasos tal como se muestra en la Figura 4.16:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos la primera cantidad (9 primeros dígitos), seguidamente dos veces el tabulador izquierdo y la tecla «ADD» para finalmente teclear los dígitos restantes y añadirlos con la tecla «ADD».
4. Repetimos el paso 3 para sumar la segunda cantidad de doce dígitos.
5. Tecleamos la tercera cantidad (9 primeros dígitos), seguidamente del tabulador izquierdo acompañado de las teclas «SUB-STOP» y «÷» para restar. La unidad restante la volvemos a restar con «SUB-STOP» y «÷».
6. Contemplamos el resultado en el registro producto (III).

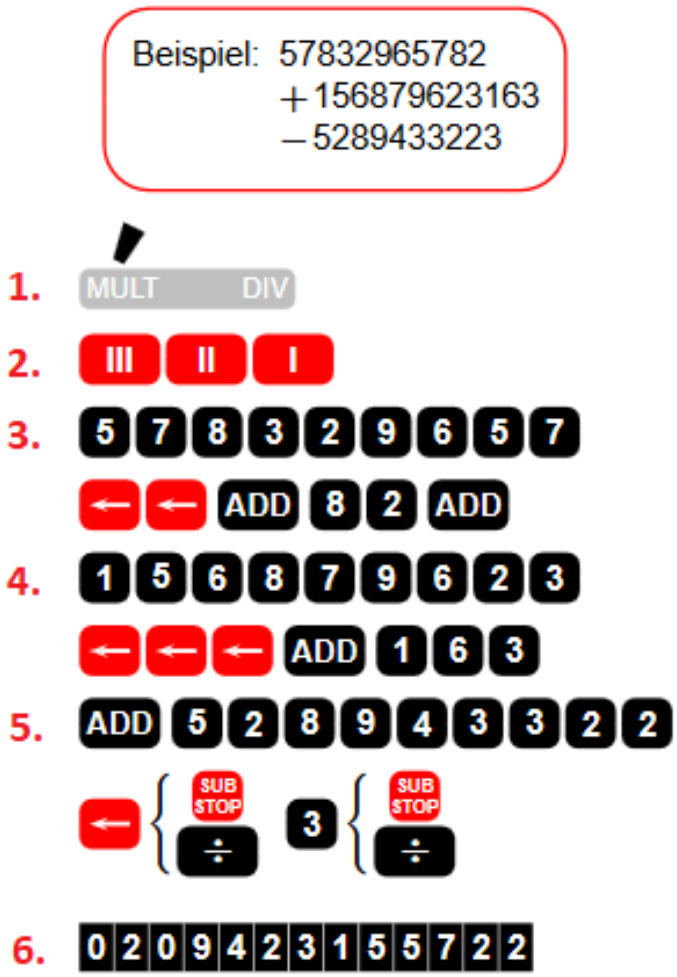

Figura 4.16: Pasos para sumar y restar cantidades de diez y trece dígitos en CA1-13. Extraído del manual CA1-13 de 1956.

Para restar cantidades negativas, seguiremos los siguientes pasos tal como se muestra en la Figura 4.17:

1. Colocamos la palanca de control deslizante en posición media.
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades, acompañada de la tecla «ADD» para después restar con «SUB-STOP» y «÷».
4. Observamos el primer resultado en el registro producto (III).
5. Para mostrar la cantidad negativa, tecleamos los últimos dígitos del primer resultado acompañado de la tecla «÷» (dos veces).
6. Observamos el resultado en el registro producto (III). OJO: El resultado que ofrece no muestra el símbolo negativo (rango numérico de 0 a 9).

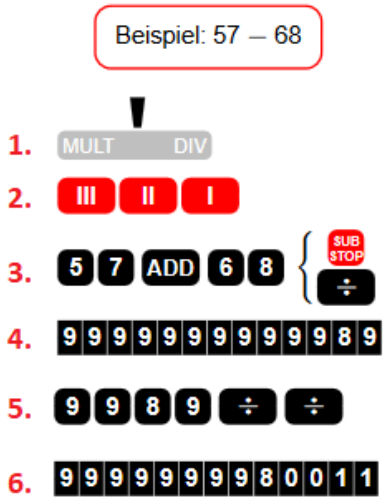

Figura 4.17: Pasos para restar cantidades negativas en CA1-13. Para ver el resultado de la operación negativa, es necesario aplicar el paso 5. Extraído del manual CA1-13 de 1956.

Para realizar una multiplicación con suma de productos, seguiremos los siguientes pasos tal como se muestra en la Figura 4.18:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades acompañadas de la tecla «X». Después de presionar la tecla igual limpiamos el registro de configuración y multiplicador.
4. Volvemos a teclear las dos cantidades restantes acompañadas de la tecla «X». Mostramos el resultado presionado la tecla igual.
5. Observamos el resultado en el registro producto (III).

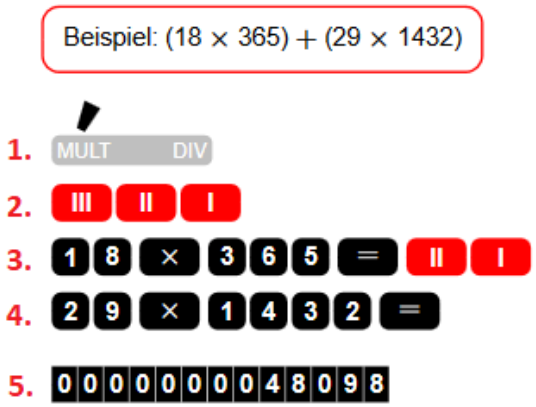

Figura 4.18: Pasos para realizar una multiplicación con suma de productos en CA1-13. Extraído del manual CA1-13 de 1956.

Para realizar una multiplicación con resta de productos, seguiremos los siguientes pasos tal como se muestra en la Figura 4.19:

1. Colocamos la palanca de control deslizante en «MULT».
2. Tecleamos las tres teclas de limpieza de registros.
3. Tecleamos las cantidades acompañadas de la tecla «X». Después de presionar la tecla igual limpiamos el registro de configuración y multiplicador para finalmente presionar la tecla «NEG».
4. Volvemos a teclear las dos cantidades restantes acompañadas de la tecla «X». Mostramos el resultado presionado la tecla igual.
5. Observamos el resultado en el registro producto (III).

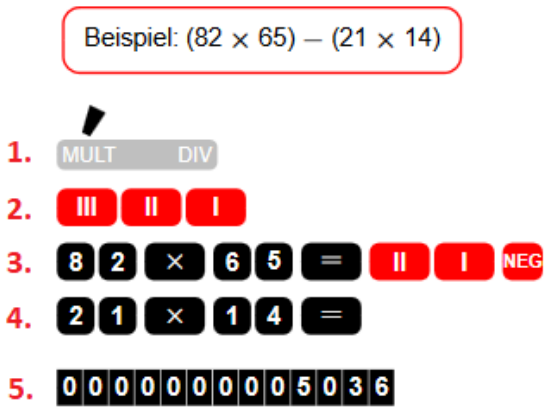

Figura 4.19: Pasos para realizar una multiplicación con resta de productos en CA1-13. Extraído del manual CA1-13 de 1956.

Para realizar una multiplicación con factor constante, seguiremos los siguientes pasos tal como se muestra en la Figura 4.20:

1. Colocamos la palanca de control deslizante en «MULT».
2. Limpiamos los registros con las tres teclas de borrado. Seguidamente, tecleamos las dos cantidades acompañado de la tecla «X», para después mostrar el resultado con la tecla igual.
3. Observamos el resultado en el registro producto (III).
4. Volvemos a teclear la tecla «X» y borramos el registro producto y multiplicador para volver a multiplicar con otra cantidad. Mostramos el resultado presionado la tecla igual.
5. Observamos el resultado en el registro producto (III).
6. Volvemos a repetir el paso 4.
7. Observamos nuevamente el registro producto (III) con el resultado.

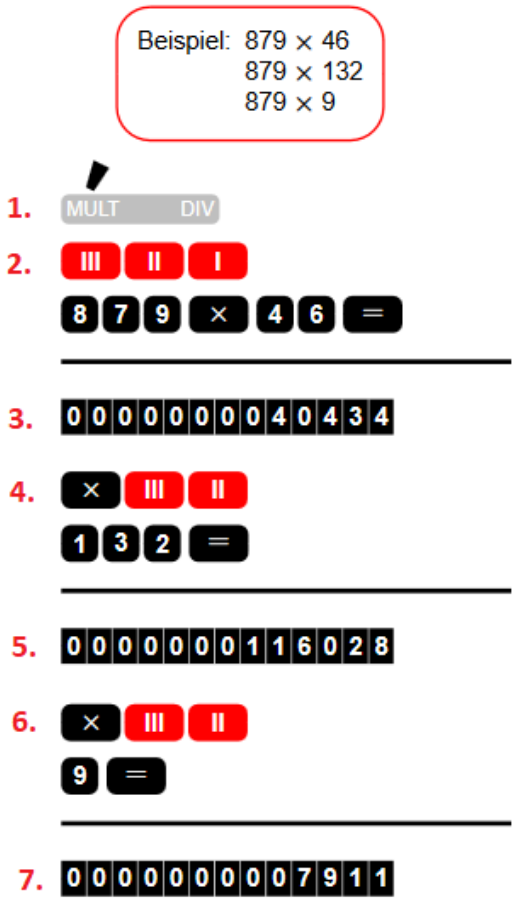

Figura 4.20: Pasos para realizar una multiplicación con factor constante en CA1-13. Extraído del manual CA1-13 de 1956.

Para agregar la coma en divisiones con decimales, seguiremos los siguientes pasos tal como se muestra en la Figura 4.21:

1. Colocamos la palanca de control deslizante en «DIV».
2. Tecleamos las tres teclas de limpieza de registros. Después introducimos el divisor acompañado del tabulador total y la tecla «ADD».
3. Ponemos una coma en la cantidad del registro producto (III). En este caso la posición de la coma es 11.
4. Introducimos el dividendo acompañado del tabulador total.
5. Colocamos la coma al principio del registro de configuración (I). Después contamos las posiciones del rango visible del número 0,003 (este caso son 6) para finalmente sumar las posiciones de rango no visibles (2 posiciones), dando como resultado un total de 8 posiciones.
6. Restamos la posición de la coma del registro acumulador con el del registro entrada (11 - 8 = 3). Colocamos en el registro cociente la coma. Ejecutamos la división con «÷» y observamos el resultado.

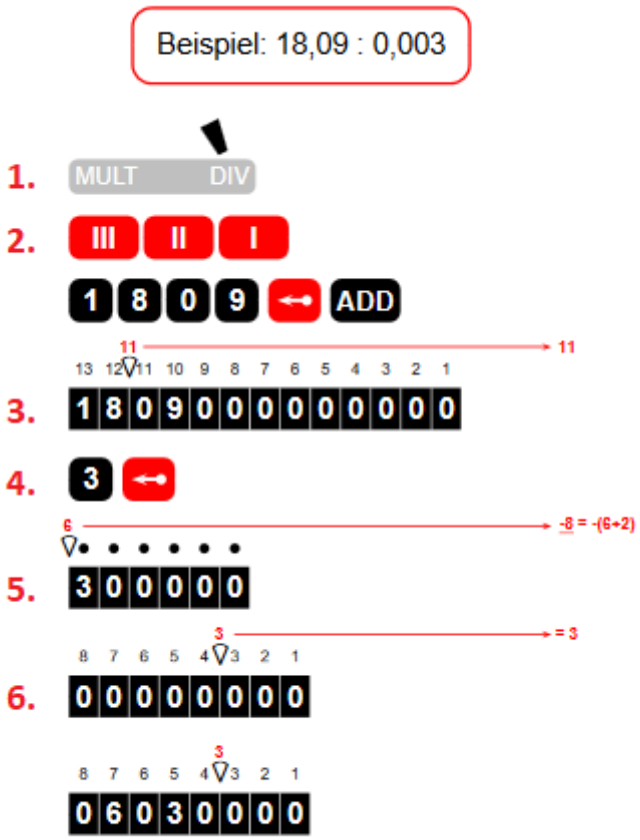

Figura 4.21: Pasos para agregar la coma en divisiones con decimales en CA1-13. Extraído del manual CA1-13 de 1956.

CAPÍTULO 5
La plataforma Scratch

El presente capítulo muestra todo lo que conlleva la plataforma Scratch, debido a que ha sido el lenguaje elegido para desarrollar la parte de implementación para este trabajo. Por ese motivo, contestaremos varias preguntas con respecto a este lenguaje de programación, tales como, ¿qué es Scratch?, ¿qué nos ofrece? y ¿por qué usarlo?, respondiendo cada una de ellas a lo largo del capítulo, mostrando una corta introducción sobre su funcionamiento, entorno y manejo del lenguaje.

5.1 Scratch, más que un juego

Hoy en día existen muchos lenguajes de programación que facilitan la comprensión del concepto «Pensamiendo Computacional». Sin embargo, no todos esos lenguajes ayudan a las personas por igual, especialmente niños y adolescente. Por ese motivo, la «Computación Creativa» es una alternativa que abarca muchas opciones de desarrollo para jóvenes como pensadores computacionales, recurriendo a perspectivas, conceptos y prácticas, a través de disciplinas y diferentes entornos de desarrollo. Una de estas herramientas especializadas en «Computación Creativa» es Scratch (Figura 5.1), ofreciendo un lenguaje de programación visual y educativa. El término visual se refiere a la manera de implementar código dentro de un entorno puzzle o montaje de piezas, ayudando a la persona a crear código de manera creativa y fácil. El usuario mientas crea el algoritmo a un lado de la ventana, en el otro lado puede verse el resultado después de ejecutar su implementación, ayudando a educadores y alumnos a comprobar el resultado, generando una retroalimentación sobre su código. Esta herramienta especializada en enseñar a programar tiene mucha efectividad en una gran variedad de colegios y talleres educativos, en donde los estudiantes aceptan ese desafío de crear y programar, mejorando así sus aptitudes antes de enfrentarse a retos más complejos [5].

El origen de la plataforma Scratch aparece por la necesidad de enseñar el «Pensamiento Computacional» a niños y jóvenes, algo que el Instituto Tecnológico de Massachusetts (MIT - Universidad de Cambridge) investigaba desde la aparición de los computadores personales. Aumentando más su interés a principios del año 2000, cuando el grupo de investigación Lifelong Kindergarten trabaja de cerca con la empresa Lego, desarrollando proyectos de colaboración en

«Robótica creativa» con legos. Durante la cooperación de las dos entidades, surgieron varios análisis y conceptos relacionados con la «Computación Creativa», moldeando la idea de un Proyecto basado en programación gráfica con bloques de código que pueden ser montados, conectados y testeados en una plataforma visual sencilla.

Figura 5.1: Logo de la plataforma *Scratch*. En un principio la mascota que ideó Resnick iba a ser un personaje de fantasía, pero por sugerencias del equipo pasó a ser un gato. Scratch viene del inglés que significa Rasguñar.

Mitch Resnick (de LLK y *Scratch Team*), John Maeda de MIT y Yasmin Kafai de UCLA propusieron la idea por primera vez a la National Science Foundation (NSF) a principios de 2003, siendo el proyecto aceptado y financiado de manera inmediata. Por esta razón ese mismo año la compañía Playful Invention Company y MIT Media Lab, desarrollan de forma conjunta la primera versión de Scratch (Figura 5.2). Desde el 2003 hasta el año 2007 el proyecto continuó su desarrollo a puerta cerrada, implementándose importantes mejoras (interfaz de usuario sencilla, aumento de funcionalidades y más bloques interactivos de código), siendo testeada por centros educativos con acuerdos exclusivos de la Universidad de Cambridge antes de ser lazado al público en 2007. Después de 2 años del lanzamiento de Scratch 1.0, el éxito de la idea se fue reflejando en el número de proyectos individuales generados por usuarios en todo el mundo (alrededor de 500.000), surgiendo una comunidad que comparte, explora y enseña el «Pensamiento Computacional» de manera fácil. Además, la plataforma continuó evolucionando hasta la versión 1.4, para después ser sustituida por Scratch 2.0 en 2013. Esta versión reescrita en Adobe Flash retoma importantes mejoras, destacando la introducción online del entorno de programación. Sin embargo, al principio surgieron varios errores en su programación y quejas de la comunidad sobre cambios en el entorno, solucionándose al poco tiempo. Finalmente, a principios del 2019, el equipo lanza la versión 3.0 escrita en HTML5 y JavaScript. En esta última versión, los desarrolladores consiguen reparar errores, logran mejorar todos sus editores instalados e integran nuevas características (más variedad de bloques y extensiones) [6, 13].

Scratch fomenta el intercambio internacional de conocimiento, apoyado siempre en las políticas de colaboración mutua e inclusión con todas las sociedades del mundo (traducido a más de 40 idiomas). Por ese motivo, se ha convertido en la última década en una comunidad con alto crecimiento de usuarios, mayormente niños de edades comprendidas entre 8 - 16 años, que invierten su tiempo en crear, desarrollar y compartir proyectos propios, para toda la comunidad.

5.2 Razones para usar Scratch

Se ha explicado muy brevemente acerca de Scratch, el origen, los motivos de aparición y sus versiones. Por tanto, vamos a mostrar de forma resumida las razones por las que se decidió utilizar esta plataforma de programación [21]:

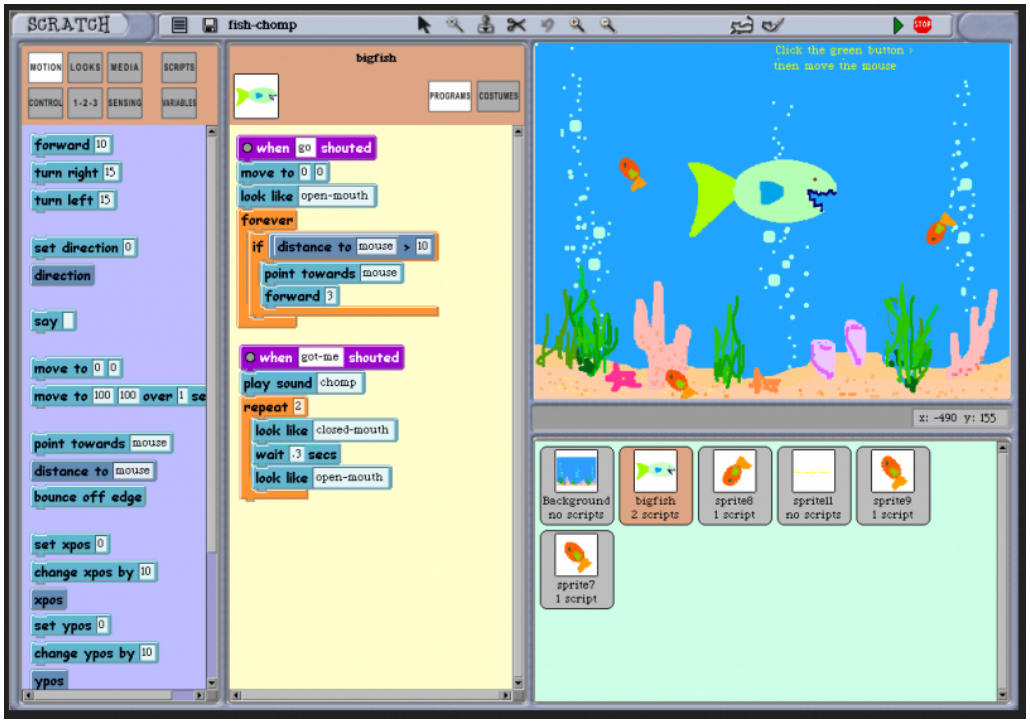

Figura 5.2: Primera versión de Scratch. El entorno genera una retroalimentación gracias al escenario donde se acoplan los objetos (*sprites*), las escenas (*stages*) y los bloques de código (*scripts*).

- **Software libre.** Scratch tiene la Licencia Pública General de GNU (GNU GPL) usada en todo el mundo del software libre y código abierto que garantiza a los usuarios finales (organizaciones, personas) estudiar, usar, compartir y modificar el software, siendo la plataforma distribuida de forma gratuita a todas las personas del mundo.
- **Sistema multiplataforma.** Es una plataforma que está disponible para diversos sistemas operativos, así como, Windows, Ubuntu, Mac. A través de su interfaz online los usuarios tienen la posibilidad de desarrollar sus proyectos en cualquier dispositivo, y en comparación con una aplicación de escritorio, evita la instalación de aplicaciones, ahorrando espacio y recursos del equipo.

- **Plataforma de alta usabilidad.** Pese a que Scratch no tenga las mismas funcionalidades que un entorno de alto nivel (compilación avanzada, uso de librerías...), no significa que fuera deficiente para el desarrollo rápido de proyectos. Como se explicó anteriormente la plataforma aparece por la necesidad de poder enseñar el «Pensamiento Computacional» a personas recien introducidas al mundo de la programación. Por tanto, es ideal para colegios, escuelas y museos (Museo de Informática de la Universidad Politécnica de Valencia) que imparten talleres de aprendizaje (*Scratch Day*).

- **Formación y aprendizaje.** El tiempo que toma para empezar a usar Scratch es relativamente corto, pudiéndose crear proyectos muy rápidamente. Por lo cual es excelente para impartir breves talleres de aprendizaje sobre algún tema en concreto, como ejemplo puede ser, el tema del presente trabajo, mostrando y enseñando de forma interactiva la historia, el uso y la implementación algorítmica de las calculadoras mecánicas Facit *C1-13* y *CA1-13*.

- **Compartir proyectos Scratch.** Al ser una plataforma de programación online, ofrece la posibilidad de compartir los proyectos que se desarrollen por internet. De esta manera, los usuarios desde cualquier lugar del mundo pueden acceder a todas las partes del proyecto (código interno, escenas y objetos usados) con la opción de realizar cambios, dando lugar a una nueva versión del proyecto Scratch.

- **Gran comunidad de usuarios.** Scratch es una comunidad de alto crecimiento de usuarios por año, por este motivo existe en internet una gran variedad de foros, talleres y bibliografías acerca de todas las versiones desarrolladas hasta el día de hoy. Permitiendo a nuevos usuarios, introducirse de forma rápida en el mundo de la programación lógica. En la Figura 5.3 tenemos estadísticas reales sobre la evolución de la comunidad en los últimos años, observando que después de la actualización a la versión 2.0 realizada en el año 2013, el número de proyectos, comentarios y nuevos usuarios crece en su mayoría de forma lineal. Además, en la actualidad cuenta con alrededor de 289 millones de comentarios enviados y 56 millones de proyectos compartidos y usuarios registrados.

- **Mejora continua de la plataforma.** Debido a la gran comunidad de usuarios, existen dentro de los foros de debates quejas, opiniones y sugerencias sobre la continua mejora de la plataforma. El equipo Scratch recibe muchas solicitudes para agregar funciones más avanzadas, como por ejemplo, programación orientada a objetos, herencia y estructuras (listas, tabla hash...), teniendo en cuenta para su futuro desarrollo e implementarlo poco a poco manteniendo siempre el enfoque principal.

5.3 Uso del entorno Scratch

5.3.1. Administrar proyectos y estudios

Después de registrarse como usuario, en el menú principal tenemos una opción *My Stuff* (Mis cosas) dentro del perfil de usuario. Al acceder podemos ver

un menú acerca de nuestros proyectos, detallando cada opción a continuación (Figura 5.4):

- **Administrador de proyectos:** Es un directorio con todos los proyectos realizados o modificados por el usuario, está dividido en «Proyectos Compartidos» y «Proyectos No Compartidos». Al gestionar algunas de estas dos opciones tenemos la alternativa de compartir o no compartir los proyectos, además de acceder para poder modificar los aspectos principales, como por ejemplo, el código (*scripts*), escenarios (*stages*) y objetos (*sprites*). Cuando se comparte algún proyecto Scratch, este pasa a tener la licencia Creative Commons pública donde cualquier usuario tiene acceso al código y elementos principales. Cuando se modifica algún proyecto hecho por otro usuario, el título lo diferencia con (*remix*).
- **Mis estudios:** Es una categoría dentro del administrador de proyectos, es utilizada para agrupar un conjunto de proyectos Scratch. En nuestro caso, para almacenar las dos calculadoras de este trabajo, asociamos con el nombre de Calculadoras Facit.
- **Papelera:** Es un archivador de proyectos borrados por el usuario, ofreciendo la oportunidad de restablecerlos. Dentro de esta categoría existe la opción de borrar definitivamente de la plataforma, dando al usuario la alternativa de liberar espacio dentro de su directorio.

Al seleccionar alguna de las categorías en la parte derecha tenemos información acerca de los proyectos (título, fecha de la útlima modificación), en donde se puede ordenar con las distintas opciones que nos ofrece. Además existen las opciones de «Ver dentro», «Borrar», utilizadas para acceder o borrar el proyecto. Por último en la parte superior derecha tenemos las opciones de crear un nuevo proyecto o un nuevo estudio («Nuevo Proyecto», «Nuevo Estudio»).

5.3.2. Entorno del editor principal

Una vez seleccionado la opción «Nuevo Proyecto» o haber abierto algún proyecto ya creado, el editor Scratch carga todas sus opciones, mostrando cuatro partes principales: la paletas de desarrollo, ubicado en la zona izquierda de la página; el área de código (*scripts*), ubicado en la parte central; la lista de objetos (*sprintes*), en la parte inferior derecha, y el escenario (*stage*), en la zona superior derecha [13]. A continuación vamos a detallar cada uno de estas áreas principales (ver Figura 5.5):

- **Paletas de desarrollo:** Es un área importante de Scratch, ya que tiene todas las instrucciones principales para desarrollar el proyecto, esta compuesto de:

  1. **Paleta de Código.** Es un conjunto de elementos utilizados para implementar scripts o programas a cada uno de los objetos seleccionados. Tiene una amplia cantidad de bloques que serán detallados más adelante.

Figura 5.3: Estadísticas sobre Scratch. Se muestra el tráfico del sitio web y las tendencias mensuales de actividad, observando un crecimiento exponencial de comentarios durante el inicio del confinamiento.

2. **Paleta de Disfraces (*costumes*).** Es una lista de los disfraces que tiene cada uno de los objetos, donde el menú ofrece las opciones de cargar, asignar, modificar, borrar y crear un disfraz. A su vez, aparece un editor de imágenes en el lado derecho, con múltiples opciones de edición. Para poder editar la imagen es necesario seleccionar la opción «Convertir a mapa de bits» ubicada en la parte inferior.
3. **Paleta de Sonidos.** Al igual que la paleta de disfraces, este está formado por una lista de sonidos que tiene asignado cada objeto, mostrando en el lado derecho un sencillo editor de sonidos.

- **Área de código (*scripts*):** Es un zona donde se van colocando los bloques de código (*scripts*) para cada uno de los objetos seleccionados. Los bloques son arrastrados desde la Paleta de Código a cualquier lugar del área de *scripts*. En la parte superior derecha se puede observar a que objeto le pertenece el conjunto de bloques de código implementado. Además en la parte inferior derecha encontramos las opciones de zoom para aumentar o disminuir la visión de los bloques.

- **Listas de objetos (*scripts*):** Esta formado por una lista de objetos (*scripts*), donde cada uno de ellos tiene una configuración única e independiente. Cada objeto esta ligado a un conjunto de imágenes o disfraces (*costumes*),

que son mayormente una variedad de la misma imagen. A su vez, se puede agregar código (*scripts*) para el desempeño de ese objeto.

- **Escenario (*stage*):** Esta compuesto de una pantalla visual, utilizada para mostrar el resultado final después de implementar los bloques de códigos a los objetos y fondos de escenario. En la parte superior derecha tenemos las opciones de «Empezar» (bandera verde), «Parar» (símbolo rojo) y en la parte izquierda tenemos «Cambio de vista» y «Expandir pantalla».

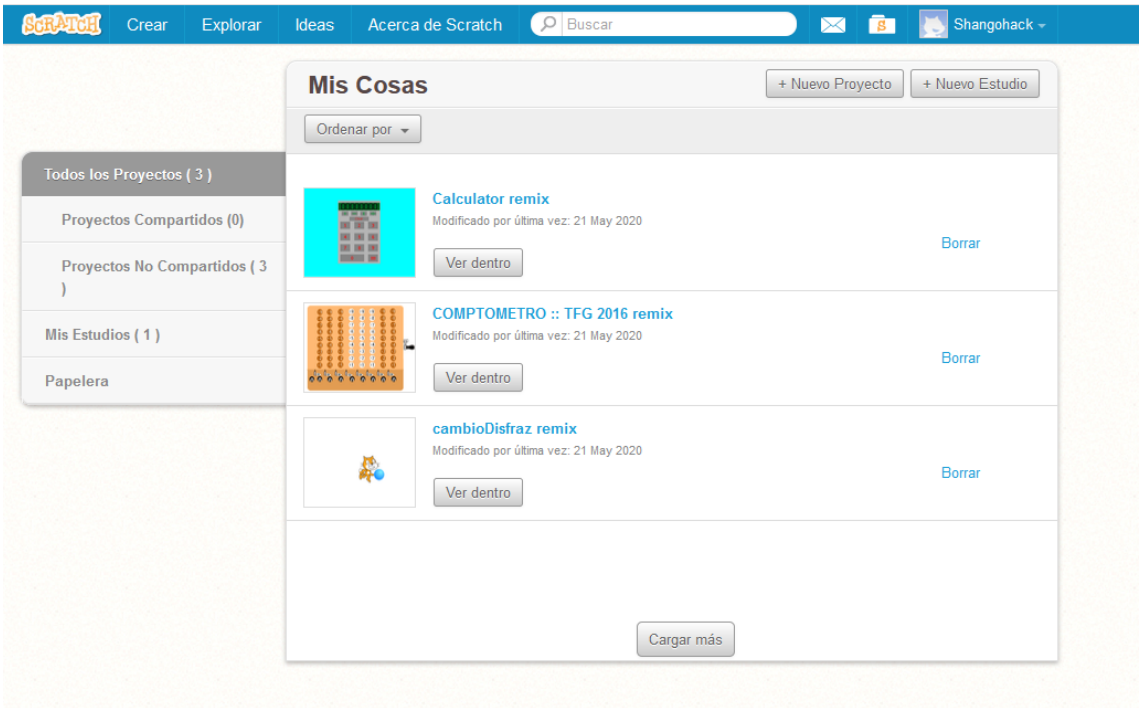

Figura 5.4: Pantalla del directorio personal en Scratch. En la cabecera tenemos un menú de opciones básicas acompañado de un buscador e inicio de sesión. En la parte central se encuentra el repositorio del usuario (Mis Cosas).

5.3.3. Construir el código

Lo más importante para un proyecto informático es la implementación del código, a través de su editor con el lenguaje elegido por el programador. En nuestro caso como se ha explicado antes, la construcción del código se realiza mediante el «montaje» de estructuras de bloques predefinidos, parecidos a piezas de lego. En Scratch, estas piezas tienen una amplia cantidad de estructuras de código, ubicadas en la «Paleta de Código» listos para ser ubicadas en el «Área de código». Seguidamente vamos a detallar los diferentes tipos de bloques predefinidos en Scratch [13].

En este momento se va a categorizar los bloques de instrucciones en dos grupos, según la silueta del bloque y después según su funcionalidad. A continución se mostrará las instrucciones según la silueta o forma que tiene el bloque (ver Figura 5.6):

- **Bloques de sombrero (*Hat blocks*):** reaccionan a una llamada dada por el usuario o a través de un evento de otro hilo en ejecución, poniendo en marcha el conjunto de código asignado a este bloque. Como ejemplo tenemos la pieza «al hacer clic en (bandera verde)» , dando la opción de arranque del programa cuando se presione la bandera verde.
- **Bloques de pila (*Stack blocks*):** esta formado por la mayor parte de instrucciones Scratch. Son órdenes dependientes de otras, y es necesario tener un «Bloque de sombrero» para que pueda ser ejecutadas. Además su ejecución es secuencial, es decir, no avanza hasta que no termina su anterior. Por ejemplo, tenemos la instrucción «preguntar y esperar» (Realiza una pregunta en el tiempo establecido).
- **Bloques de tapa (*Cap blocks*):** son instrucciones que finalizan el hilo de ejecución de un conjunto de código. Como dos únicos ejemplos tenemos, «detener» (detiene la ejecución), y «borrar este clon» (borra la instancia de un objeto).

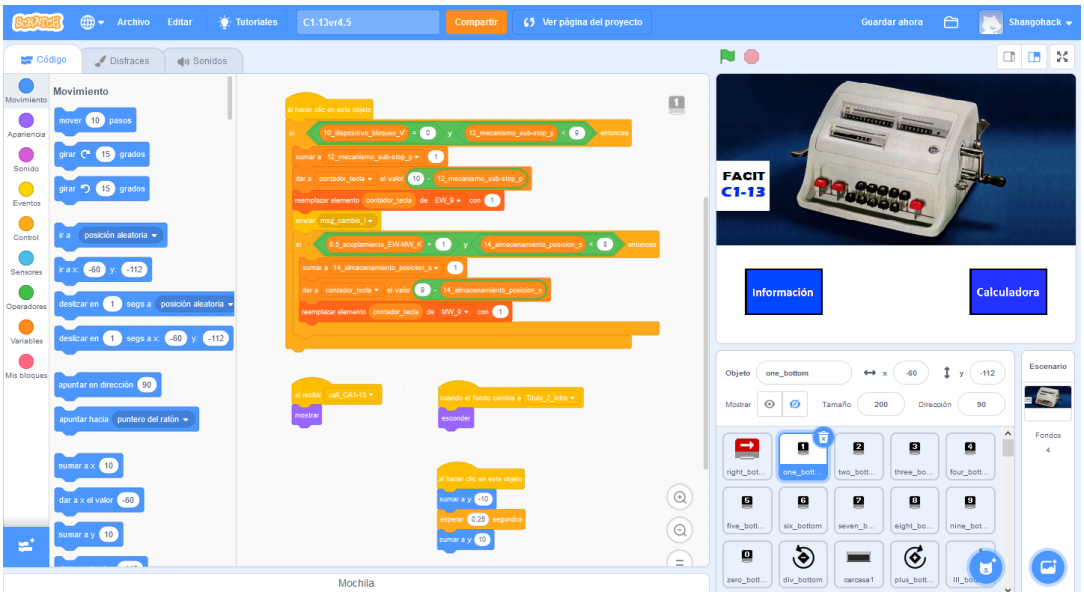

Figura 5.5: Pantalla del editor principal de Scratch.

- **Bloques C (*C blocks*):** son instrucciones con forma de C, utilizados para el flujo de control de un programa (condiciones y bucles). Entre los ejemplos comunes tenemos el bloque de condición «si...entonces» (condición if-else) y el bloque de repeticiones «por siempre» (bucle de repeticiones infinitas).
- **Bloques de variables (*Reporter blocks*):** instrucciones con forma ovalada, son utilizadas para almacenar variables numéricas o cadenas de caracteres (*strings*). Se pueden emplear para agregar una variable a cualquier otro bloque, ayudando a desempeñar alguna acción de forma dinámica. Como ejemplos tenemos «nombre de usuario» (variable predefinida que devuelve el nombre de usuario), o cualquier variable determinada por el programador.
- **Bloques booleanos (*Boolean blocks*):** están representados con forma hexagonal, son utilizados para operaciones booleanas con respuesta de «true» o

«false». Al igual que los bloques de variables, se puede utilizar dentro de otro bloque para generar una acción de manera lógica. De ejemplo tenemos «¿tocando el color?» (devuelve true si el objeto toca algo de ese color), o cualquier operador lógico «a and b» (agregando otros bloques booleanos a los parámetros a, b).

Continuando con la descripción del segundo grupo, vamos va a detallar los bloques según su funcionalidad (ver Figura 5.7):

Figura 5.6: Conjunto de bloques según su silueta Scratch.

- **Bloques de movimiento (*Stack blocks*):** son bloques de color azul, usados para dar movimiento a un objeto en la escena. En esta categoría tenemos; desplazamientos basados en las posiciones «x» e «y» y rotación según su ángulo.

- **Bloques de apariencia (*Looks blocks*):** son bloques de color morado, utilizados para dinamizar la apariencia de cada objeto. Dentro de esta clase tenemos la representación de mensajes por pantalla, variables de información sobre un elemento (disfraz, fondo, objeto) y funciones de aspecto dinámico (cambio/desaparición de disfraz, fondo u objeto dentro del escenario).

- **Bloques de sonido (*C blocks*):** son bloques de color púrpura, usados para dar un sonido específico a un objeto determinado. Aquí tenemos la opción de iniciar un sonido, dinamizar el efecto del sonido, modificar el volumen, detectar y quitar sonido.

- **Bloques de eventos (*Event blocks*):** son bloques de color amarillo, que permiten la ejecución de un conjunto de código al recibir un evento (mensaje). Además dentro de esta categoría tenemos los bloques de enviar mensaje (con y sin tiempo de espera).

- **Bloques de control (*Control blocks*):** son bloques de color naranja, usados para controlar el flujo del programa. Como se detalló anteriormente, esta compuesto en condiciones y bucles. Entre las condiciones tenemos; «si...entonces», «si...entonces...si no» y para los bucles esta; «repetir», «por siempre», «repetir hasta que». Asimismo, tenemos las funciones «esperar», «esperar hasta que» y «detener».

- **Bloques de sensores (*Sensing blocks*):** son bloques de color cian (azul marino), que permiten detectar si algún elemento (teclado, ratón, objetos (*sprints*)) es utilizado o realiza alguna acción. Entre la detecciones que realiza tenemos: el tocar otro objeto o color, entrada de datos por teclado, uso del ratón o teclado, distancia entre objetos.

- **Bloques de operadores (*Operator blocks*):** son bloques de color verde, usados para realizar operaciones aritméticas (*suma, resta, multiplicación, división, módulo y valor absoluto*), operaciones lógicas (*and, or, not*) y comparadores (*mayor que, menor que, igual*). Además tenemos funciones como concatenar cadenas de caracteres, sacar la letra de la posición de una cadena y búsqueda de un carácter en un cadena.

- **Bloques de variables (*Data blocks*):** son bloques de color naranja oscuro, que permite declarar variables y listas (*arrays*) de tipo númerico o alfanumérico. Al crear una variable o lista, nos lanza la opción de hacerla global o local. Además en esta categoría tenemos varias funciones como; dar, sumar, mostrar y esconder un valor. Y para una lista tenemos; añadir, eliminar, insertar y reemplazar elementos. Asimismo, tenemos las opciones de obtener el número de elementos, la posición del elemento, longitud de la lista y búsqueda de una variable en una lista.

- **Mis bloques (*My blocks*):** es un bloque de color rosa, utilizados para crear funciones personalizadas por el programador para un objeto (no se puede realizar llamadas entre objetos). Este bloque tiene la opción de asignar parámetros de entrada (numérica, texto y booleana).

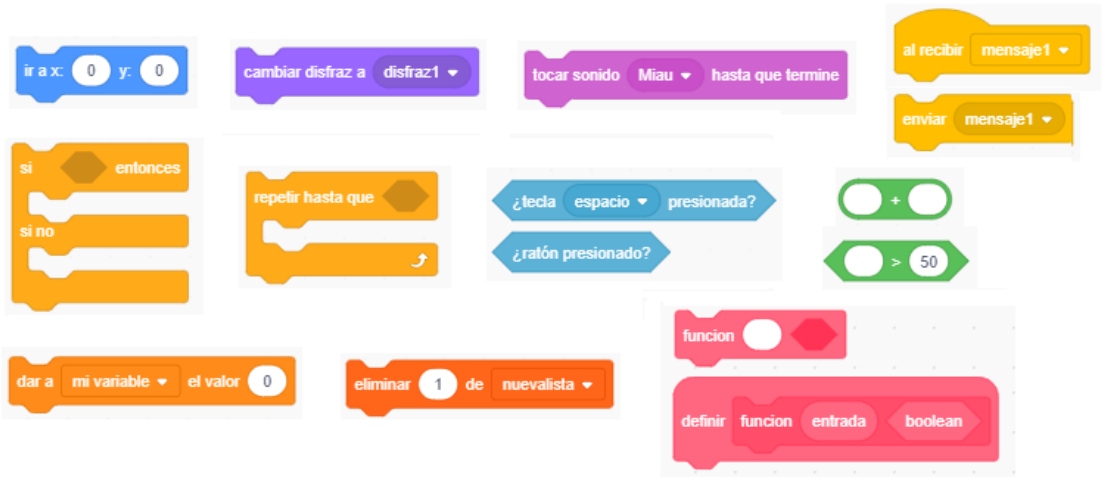

Figura 5.7: Conjunto de bloques según su funcionalidad Scratch.

CAPÍTULO 6
Diseño e implementación con Scratch

El actual capítulo presenta una descripción del análisis, diseño e implementación de los dos proyectos planteados para este trabajo. Se expone primeramente el análisis y diseño general de los proyectos mediante diagramas, detallando las funcionalidades comunes entre ambos, para después mostrar de forma individual las partes específicas de cada modelo simulado. Las dos aplicaciones desarrolladas que serán incorporadas a la web del Museo de Informática, ofrecen una visión interactiva, recreativa y totalmente visual al usuario. Por último se muestra la capacidad que puede ofrecer la herramienta Scratch para la creación de aplicaciones interactivas, analizado y diseñado desde la extensa seriedad de un proyecto software profesional.

6.1 Diseño general de los proyectos

6.1.1. Análisis software

Antes de empezar a desarrollar los proyectos es necesario determinar el análisis que debe tener una aplicación, tal y como la funcionalidad, la interfaz y el diseño, mostrando una visión general de ambos proyectos, detallando en sí sus similitudes. Empezamos con la realización de los requisitos funcionales a través del diagrama de «Casos de Uso», permitiendo expresar de forma esquematizada y simplificada el comportamiento de la aplicación ante los usuarios finales. Dentro del diagrama tenemos personajes o entidades (llamados también actores) que interactúan con un caso de uso. Un caso de uso es una secuencia de interacciones que ejecuta órdenes dentro del sistema, en respuesta a un evento que inicia el actor. A su vez tenemos dos tipos de relaciones: «extend» usado para representar una extensión de una funcionalidad principal, modelando una parte opcional del sistema; «includes» usado para enriquecer los casos de uso conectados por esta relación y a su vez compartiendo la funcionalidad común. Estos diagramas sirven para mostrar el comportamiento y la comunicación del sistema por medio de la interacción que realiza el usuario, permitiendo esta técnica concentrar la atención al analista en las necesidades del usuario final.

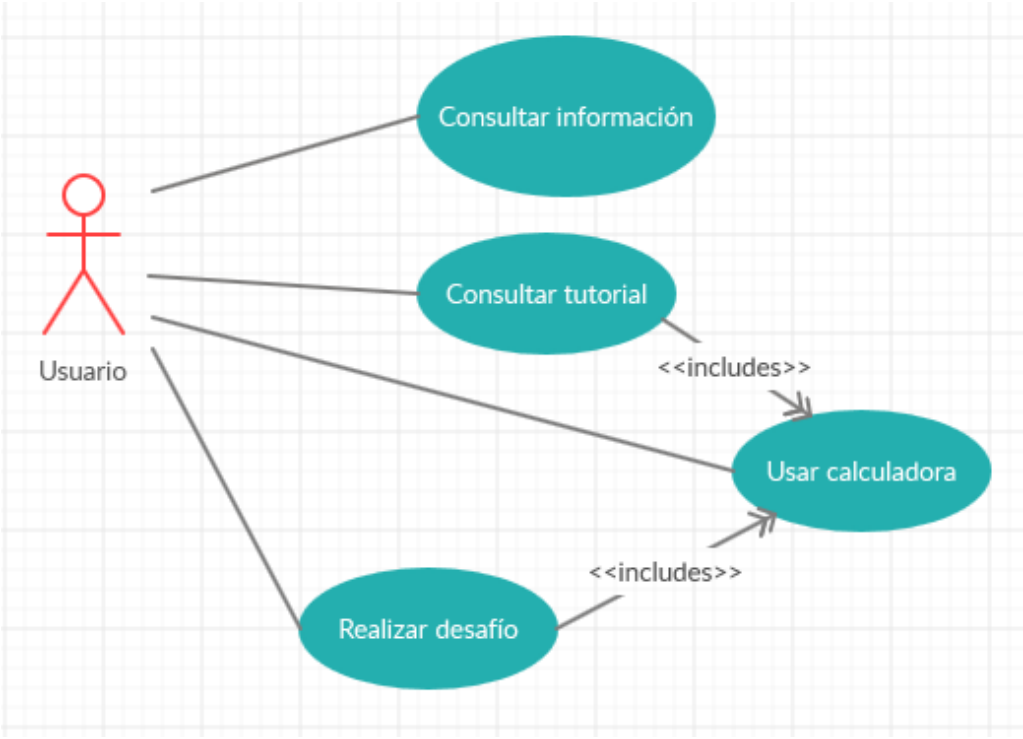

Figura 6.1: Diagrama de Casos de Uso. Compuesto de cuatro casos de uso y un actor, mostrando al usuario el diseño que tendría antes de ser implementado.

En la Figura 6.1 se muestra el Diagrama de Casos de Uso de ambos proyectos, en donde tenemos los siguientes componentes:

- **Usuario (*Actor*):** es la persona o entidad que interactúa desde fuera del sistema, en la cual demanda una funcionalidad específica. Está conectado con cuatro funcionalidades (Casos de uso): Consultar información, Consultar tutorial, Usar calculadora y Realizar desafío.
- **Consultar información (*Caso de Uso*):** Es un componente funcional en la cual el usuario tiene la oportunidad de consultar la información general de la calculadora específica, mostrando al usuario una breve introducción de forma descriptiva.
- **Consultar tutorial (*Caso de Uso*):** Es un componente funcional en donde el usuario tiene la oportunidad de aprender a utilizar de manera interactiva el modelo de calculadora específica. Este componente tiene una relación (*includes*) con «Usar calculadora» ya que para poder seguir el tutorial es necesario usar el simulador.
- **Usar calculadora (*Caso de Uso*):** Es un componente funcional que muestra la emulación de un modelo de calculadora específica. Este componente se explicará más adelante dentro de este capítulo.
- **Realizar desafío (*Caso de Uso*):** Es un componente funcional en donde el usuario, mediante la calculadora, realiza un desafío con respecto a la utilización del simulador. Al igual que el componente «Consultar tutorial», es obligatorio tener el simulador para ejecutar el desafío.

6.1.2. Diseño visual

Después de tener las funcionalidades principales del proyecto software, a continuación detallaremos el diseño visual, dando al lector varias muestras reales de la aplicación terminada, haciendo hincapié en cada una de estas funcionalidades. Como se describió anteriormente, la aplicación está hecha para un público juvenil. Por ese motivo, la parte visual es de suma importancia para la interacción del usuario con la aplicación. Para el desarrollo del diseño visual se utilizaron varios editores como por ejemplo GNU Image Manipulation Program (GIMP)[1], el editor web Piskel[2] y el editor que incluye Scratch, ayudando a modificar y crear fondos de presentación, botones, objetos visuales de representación y rótulos de texto. Debido a que Scratch ofrece la oportunidad de importar imágenes, resulta fácil desarrollar figuras y elementos con instrumentos de diseño externas para después integrarlas a Scratch. De esta forma, se ahorra tiempo en el desarrollo de imágenes, ya que ambos proyectos comparten muchas similitudes a la hora de presentar visualmente al usuario.

Figura 6.2: Página del menú general con sus tres opciones (izquierda) y una página informativa con datos relevantes de la calculadora (derecha).

Como se describió anteriormente, el diseño visual es muy similar para las dos aplicaciones, aunque cuenta con varias diferencias en la forma, el color y la distribución visual de cada uno de los elementos principales y secundarios. A continuación se mostrará la estructura que forma parte del diseño visual principal de los proyectos:

- **Página de presentación:** Al ejecutar el proyecto, el sistema muestra la primera escena con los créditos iniciales compuestos por: el nombre del proyecto, tutor, autor y los tres logos representativos de la universidad (Escuela Técnica Superior de Ingeniería Informática, Universidad Politécnica de Valencia y del Museo de Informática).
- **Menú (Figura 6.2):** después del lanzamiento de la página de presentación, el sistema muestra las diferentes opciones funcionales de la aplicación. Los

[1] Aplicación descargada de: http://www.gimp.org.es/
[2] Herramienta online usada en: https://www.piskelapp.com/

menús están formados por fondos y gráficos sencillos, mostrando en la parte superior central la imagen de la calculadora específica y en la parte inferior las distintas opciones.

- **Páginas informativas (Figura 6.2):** son escenas compuestas de fondos sencillos con texto descriptivo, que ofrece información específica de la calculadora. La secuencia de las escenas son de interacción sencilla, es decir, en la parte inferior de la página, el usuario tiene la opción de avanzar o retroceder la escena.

- **Simuladores (Figura 6.3):** para la representación del simulador en concreto, se trata de seguir la estética tradicional de los modelos calculadores típicos Facit. Por ese motivo, tanto el modelo *C1-13* como el *CA1-13* comparten la misma forma, color y tamaño de carcasa. Además sus teclas principales, los registros, punteros y comas deslizables son exactamente iguales. La diferencia que tienen, es sobre la colocación de los objetos dentro de cada simulador, es decir, para el modelo *C1-13* las opciones de borrado para los registros son representados por palancas, ubicados en los lados extremos de la carcasa. En cambio para el modelo *CA1-13* todas sus teclas están ubicadas en la parte frontal de la calculadora. Sin embargo, dentro del simulador existen dos escenas especiales, «Tutorial» y «Modo Reto». Estas escenas tienen el simulador acompañado de un personaje orientador (Facitman), que ofrecen una guía al usuario de forma específica.

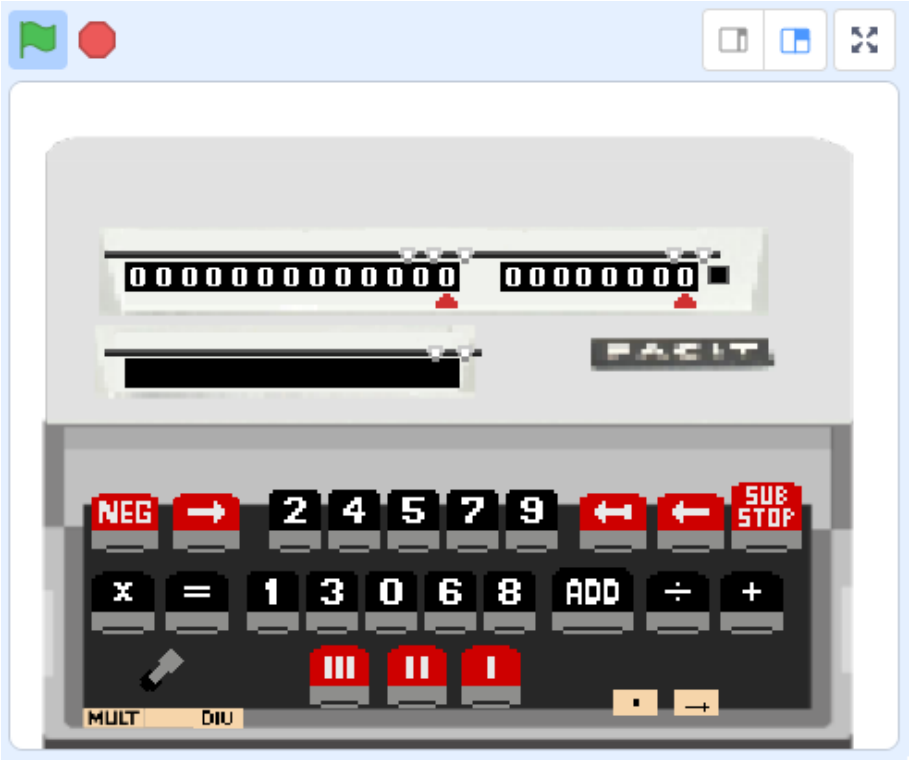

Figura 6.3: Ejemplo visual de uno de los simuladores implementados. En este caso se trata del modelo CA1-13 en su «Modo libre».

6.1.3. Navegación e interacción

El control de flujo hacia las distintas escenas o páginas del programa (navegación del programa), es exactamente igual para los dos proyectos. Por esa razón, la implementación principal de esta característica se basa en la llamada de eventos a través de mensajes, sincronizando los objetos con el escenario.

Por ejemplo (ver Figura 6.4), al ejecutar el proyecto (presionar la bandera verde) nos aparece la pantalla de presentación, quedándose fija durante tres segundos para después dar paso al siguiente fondo. Observándose el menú general, donde el usuario tiene la capacidad de interactuar con las opciones que le ofrece el menú. Esta conducta es implementada con un bloque de inicio de evento «al hacer clic en (bandera verde)», cambiando el fondo que divulga el mensaje durante dos segundos (bloque esperar) para finalmente realizar una transición al siguiente fondo con la aparición de dos objetos, teniendo como resultado el menú general (ver Figura 6.2). Evidentemente, el pequeño conjunto de código no es apreciable para el usuario, considerándose capaz de emitir un gran número de órdenes dentro del sistema. De este modo, si otros objetos de la aplicación se definen con uno o más bloques, este acoge algún evento, logrando la sincronización de estos objetos, implementando fácilmente el cambio de escenas.

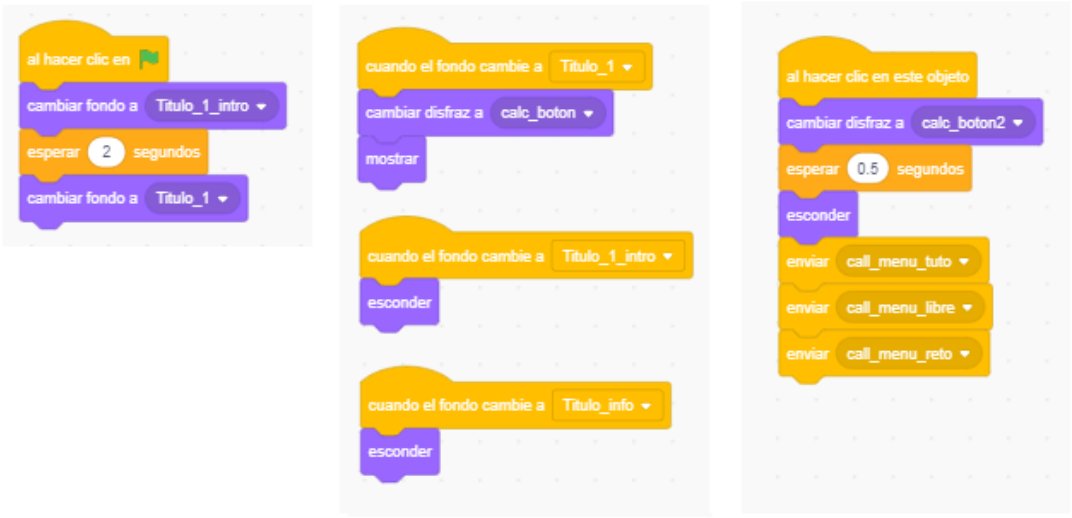

Figura 6.4: Código de la navegación básica. El bloque izquierdo muestra la página de presentación, realizando después un cambio de fondo. El conjunto central de bloques muestra el botón «Calculadora» en el menú, escondiendo el objeto cuando se realiza un cambio de fondo. El bloque derecho realiza una animación al botón para después esconderlo y finalmente envía tres mensajes para activar los siguientes botones del submenú.

Continuando con el ejemplo, al aparecer el nuevo fondo del menú general, los objetos se manifiestan con el bloque de evento «cuando el fondo cambie a», ejecutando a su vez una instrucción de cambio de disfraz para ese objeto. Cada uno de los objetos mostrados en el menú tiene la instrucción de cabecera «al hacer clic en este objeto», brindando la oportunidad de ejecutar una nueva secuencia de código. Al presionar el objeto, éste ejecuta las instrucciones de cambio de disfraz y esconder objeto, para después dar paso a un cambio de escena. En la Figura 6.4 se muestra la parte del código de forma global, observando la sincro-

nización orientada a eventos. La navegación restante entre distintas escenas para los dos proyectos sigue una temática similar, ofreciendo una navegación sencilla a través de los botones del menú o submenú. Hay que indicar que el recomienzo del proyecto provoca un reinicio de todas las variables, evitando la aparición de comportamientos incoherentes dentro de la navegación o el simulador.

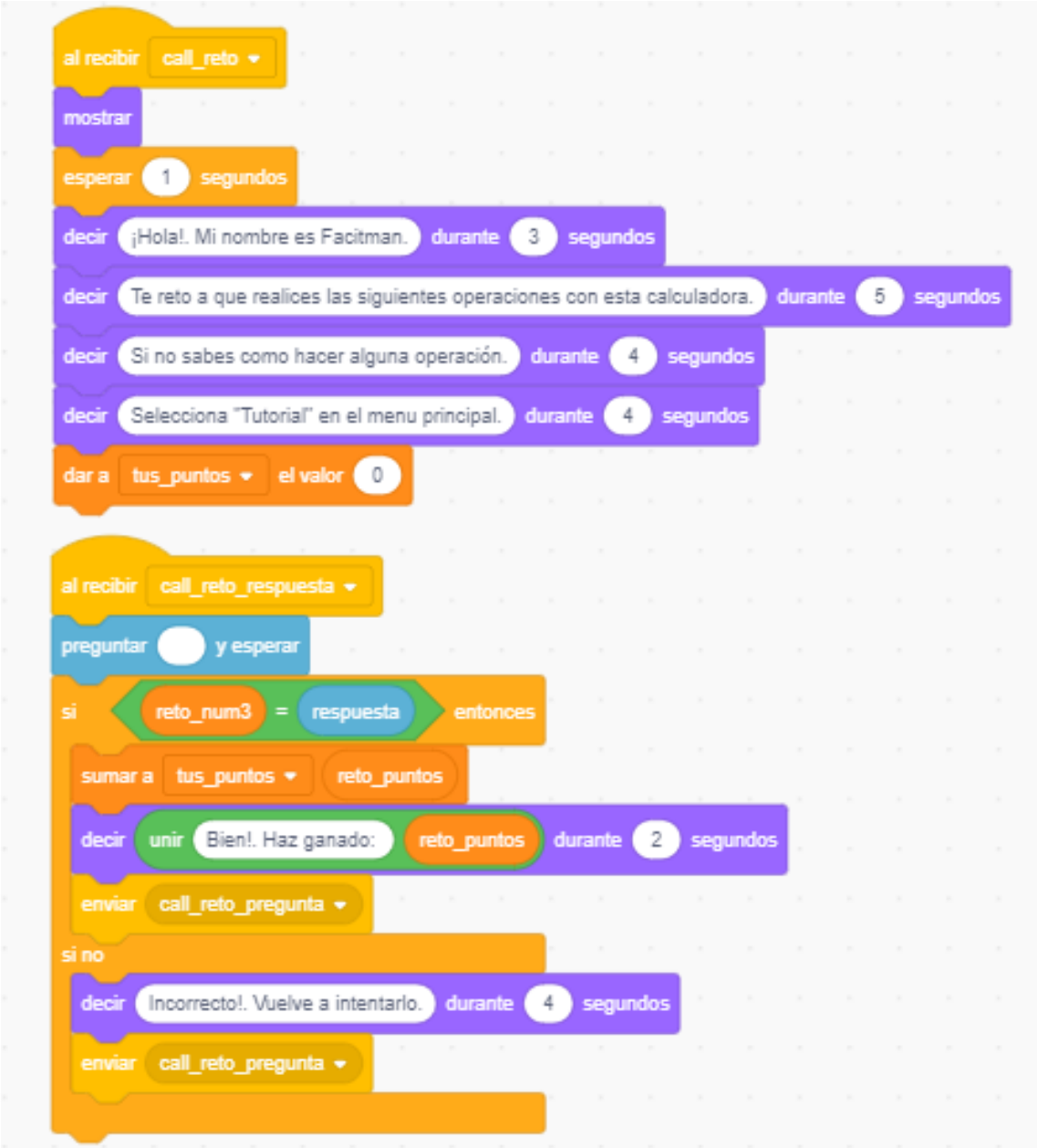

Figura 6.5: Código del caso de uso «Realizar desafío». El primer bloque genera una animación sobre el objeto, mostrando al usuario carteles de lo descrito en cada bloque violeta. Además este tipo de estructura es utilizado para el caso de uso «Consultar tutorial». El segundo bloque es el encargado de recibir la respuesta dada por el usuario.

Una vez expuesta de forma general la navegación que tienen los proyectos, pasamos a detallar la interacción que tiene el usuario con la aplicación, centrándose especialmente en el objeto «Facitman». Este objeto es muy importante ya que ofrece la funcionalidad a los casos de uso «Consultar tutorial» y «Realizar desafío». El objeto «Facitman», a diferencia de la navegación del menú, lleva en

su mayoría animaciones y cambios de información visual con el usuario. Este tipo de implementación está basado en el comportamiento al hacer clic en un objeto. Por tanto, para no volver a repetir el análisis del código anteriormente descrito, se explicará solo el concepto con dos ejemplos relacionados a la interacción de los caso de uso en particular («Consultar tutorial», «Realizar desafío»).

Después de seleccionar la funcionalidad «Consultar tutorial», el sistema muestra visualmente la calculadora acompañado del personaje «Facitman». El personaje inicia su presentación al usuario para poco después ser un «guía» en la utilización del simulador. Además, el usuario tiene la opción de seleccionar una y otra vez las opciones mostradas. Este concepto es el mismo para la animación del personaje, es decir, realizamos clic en cualquier botón, para después ejecutar secuencialmente en intervalos cortos de tiempo varios cambios de disfraz al objeto relacionado con el personaje «Facitman», mostrando al usuario los pasos que tiene que seguir con el fin de realizar una operación con el simulador.

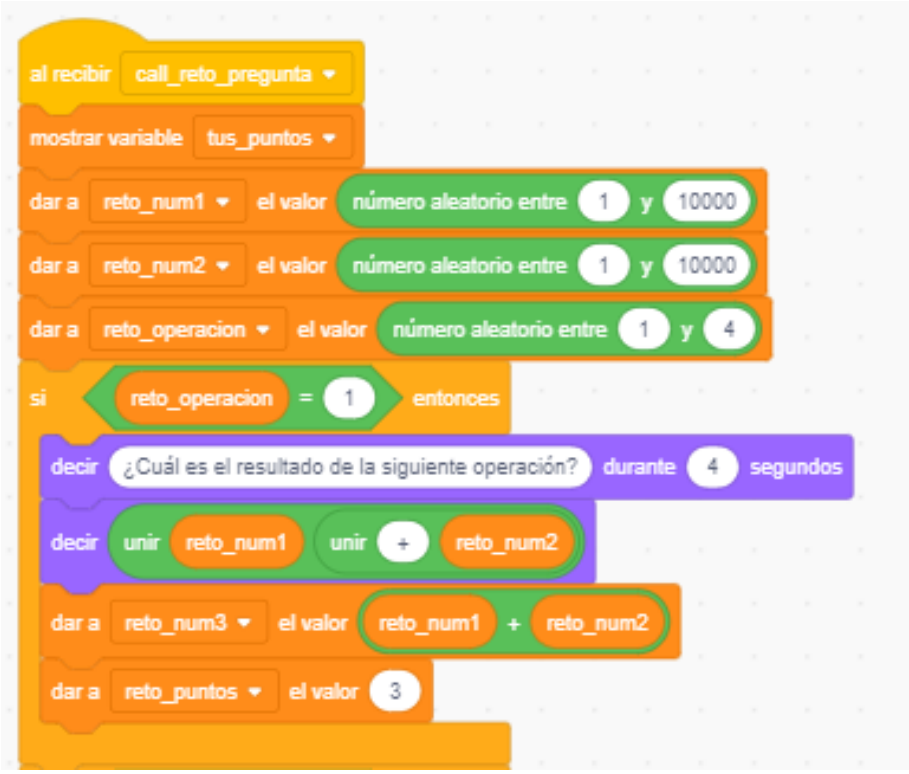

Figura 6.6: Tercer bloque del caso de uso «Realizar desafío» (incompleto). Genera una pregunta operacional aleatoria a partir de dos números con rango de 1 a 10000.

La lógica interna y el funcionamiento del personaje son exactamente iguales en todos los proyectos. Adicionalmente, el caso de uso «Realizar desafío» tiene una configuración diferente para la interacción con el usuario (ofrece y pide respuesta, analiza y genera puntos). Por ese motivo, fue necesario realizar una implementación interactiva diferente al caso de uso «Consultar tutorial», pero siempre basado en la animación y cambio de disfraz.

En la Figura 6.5, se muestran las partes principales de la implementación usada para el caso de uso «Realizar desafío». Como se puede observar en el primer bloque superior, al recibir el mensaje *call_reto*, el objeto «Facitman» es mostrado para poco después en pequeños intervalos de tiempo generar sus mensajes

al usuario, y finalmente reiniciar el valor *tus_puntos* antes de dar comienzo al desafío. A continuación, con el segundo bloque, después de recibir el mensaje *call_reto_respuesta* (cuando se da clic en el botón «Empezar») el sistema lanza una casilla para introducir el resultado, analizando la respuesta con el valor *reto_num3* (valor generado al realizar una operación aleatoria con los valores *reto_num1* y *reto_num2*). Poco después de analizar el resultado, los mensajes informativos son detallados al usuario por el personaje «Facitman». Finalmente, el último bloque (ver Figura 6.6) es utilizado para generar de forma aleatoria una pregunta con su respuesta, siendo después analizada por el segundo bloque. La implementación genera números aleatorios, asignando a tres tipos de valores (dos operacionales y un valor para el tipo de operación). En función del tipo de operación generado por números aleatorios del 1 al 4, producirá un mensaje preguntado al usuario la respuesta de esa operación.

Figura 6.7: Bloque principal para el inicio de variables. En este ejemplo tenemos distintas variables con nombres intuitivos, que generan una reacción u otra en el simulador cuando se asignan un valor distinto. En este caso la calculadora está en modo «Stand-by»

6.1.4. Lógica

En capítulos anteriores se detalló el mecanismo y utilización de cada una de las calculadoras, explicándose específicamente los modelos *C1-13* y *CA1-13*, dando a entender que ambos modelos tienen muchas similitudes dentro del mecanismo lógico operacional (añadir números, representar cantidades y operar cálculos). Por este motivo, en este apartado se estudiará la implementación lógica común y más relevante, acerca del comportamiento de los simuladores. Para los dos casos, el código que define la lógica está dividido en una «Configuración inicial» y tres bloques lógicos (Motor, Rotor y mecanismo_rotación_cociente), ubicados dentro del objeto «carcasa». A continuación vamos a detallar cada uno de estos:

- **Configuración inicial (Figura 6.7):** Este bloque de código es utilizado para regular una configuración única dentro de la calculadora (suma, resta, multiplicación, división) antes de lanzar a ejecución el proceso de cálculo. En el ejemplo, podemos observar que la configuración establecida genera un estado de «Stand-by» o modo de espera. La programación basada en flujo mediante estados, resulta una solución factible para generar una fluidez dentro del programa; más adelante, de forma específica, explicaremos cada uno de estos estados en relación a las dos calculadoras implementadas.

- **Bloque Motor (ver Apéndice A):** Este bloque de código es el encargado de redirigir el flujo de un estado predefinido por el usuario. El código es relativamente extenso ya que el lenguaje Scratch no ofrece un alto nivel para programación de condiciones, generando un bloque largo y poco entendible. Por ese motivo en el Anexo, en forma de «Pseudocódigo» estilo Javascript, podemos observar el código completo de este bloque. Continuando con la explicación, el bloque Motor tiene varios estados programados para la realización de operaciones, entre estos tenemos: Suma, Resta, Multiplicación automática, Opción de parada (Sub-Stop), División automática, Multiplicación semi-automática y División semi-automática. Por tanto, dependiendo de la configuración inicial realizada por el usuario, este bloque obliga al sistema a realizar una operación u otra.

- **Bloque Rotor (ver Apéndice A):** Este bloque es el encargado de realizar las distintas operaciones definidas en el sistema. Una vez que el usuario determina un estado y el «Bloque motor» lo detecta, el flujo se desplaza a este bloque responsable de realizar las operaciones. Su programación está hecha especialmente de dos vectores (*arrays*), uno principal (RW_13) y otro auxiliar (RZ_13). El array principal es el encargado de generar el resultado final para el «Registro Producto» (13 dígitos) y el array auxiliar es utilizado para agregar en cada posición calculada un valor en función de si el número resultado de la posición del array principal, es un dígito perteneciente al rango 0 a 9 (si está en este rango se le asigna un 0 en caso contrario un 1), para después volver a recalcular la cantidad del array principal debido a que el vector solo permite en cada posición unidades positivas.

- **Bloque mecanismo_rotación_cociente (ver Figura 6.8):** Al igual que el «Bloque Rotor», este tipo de bloque es utilizado para generar la respuesta en el «Registro Multiplicador». En este caso, los dos arrays están compuestos de

distinto tamaño, debido a que la longitud del «Registro Multiplicador» solo permite 8 dígitos, teniendo como array principal (QW_8) y el array secundario (QZ_7). Por tanto el funcionamiento e implementación es similar al «Bloque Rotor». Para evitar repetir la explicación del funcionamiento, se va a mencionar que la implementación es similar pero con pequeñas diferencias a la hora de generar el resultado (longitud de los registro del array).

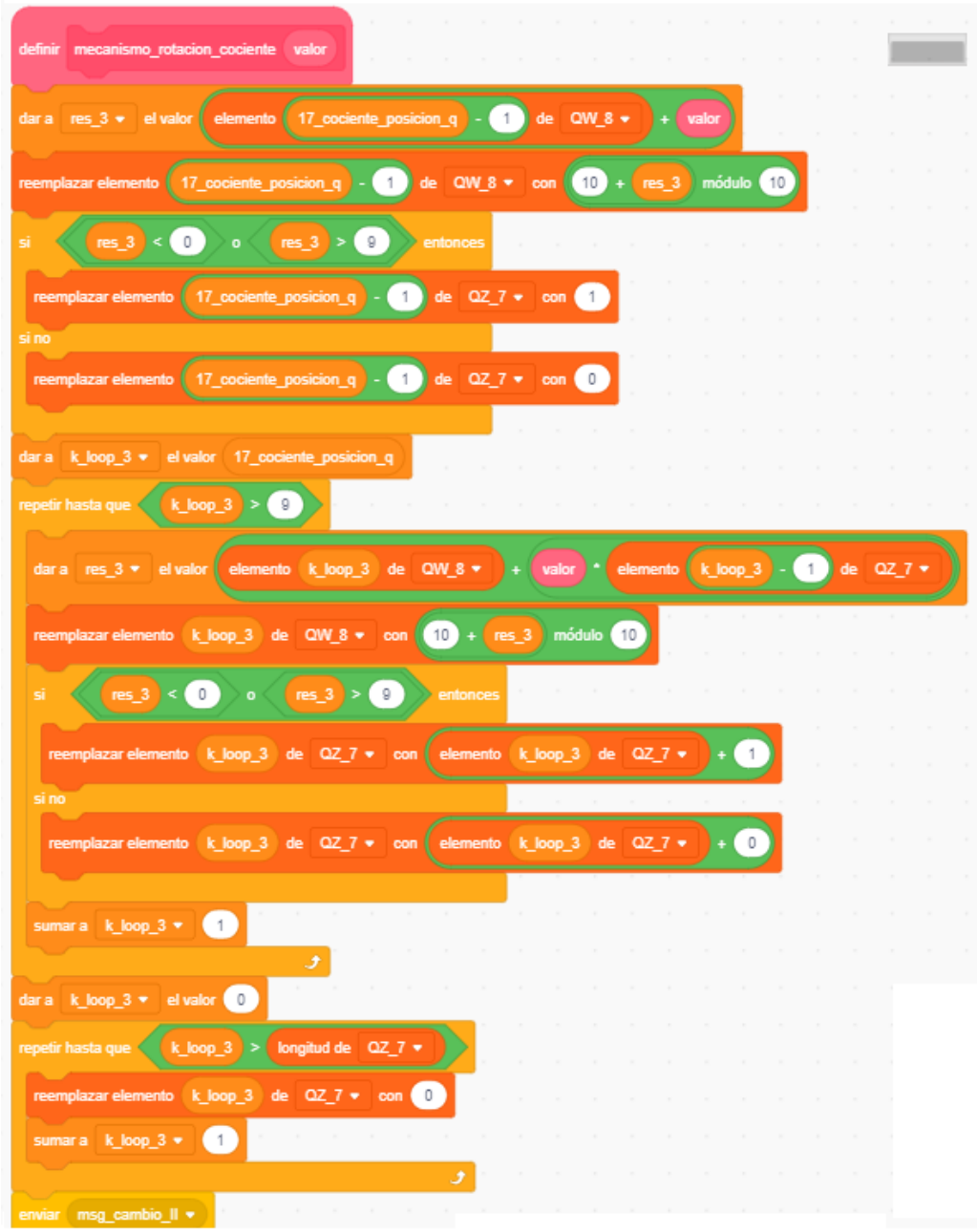

Figura 6.8: Fragmento del Bloque mecanismo_rotación_cociente. Utilizado para realizar los cálculos al registro multiplicador, está compuesto de dos arrays. Array principal QW_8 y un array secundario QZ_7. La estructura es muy similar al «Bloque rotor». El resultado se almacena en el array principal QW_8.

6.1.5. Otros componentes comunes

Hasta el momento se han descrito las partes comunes de los proyectos como: el análisis, los elementos de navegación, interacción y lógica. Sin embargo, existen otros elementos que comparten similitudes dentro del simulador, tales como las teclas, tabuladores, registros, punteros, comas deslizables y la señal de la dirección de rotación. En este último apartado vamos a describir brevemente cada uno de ellos, mostrando a su vez la implementación realizada.

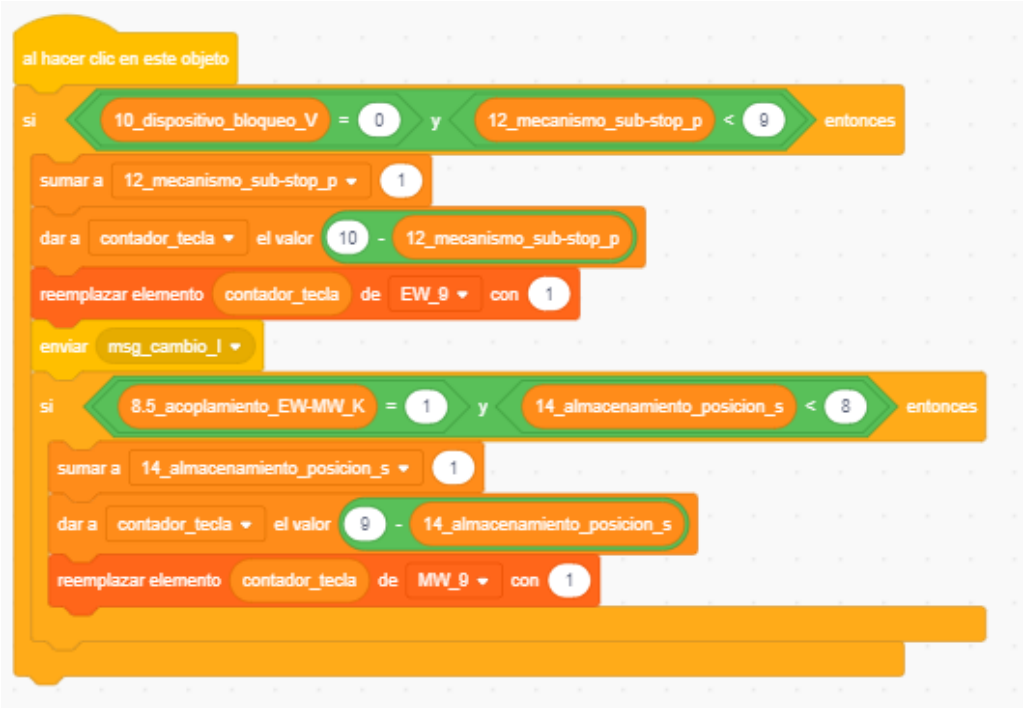

Figura 6.9: Fragmento de código de la tecla numérica 1. El bloque tiene un sistema de bloqueo, evitando incosistencias o «bugs» a la hora de realizar algún tecleo erróneo.

Antes de empezar con cada uno de los componentes comunes, hay que aclarar que los simuladores tiene un sistema de bloqueo general. Este sistema de bloqueo impide al usuario continuar con la ejecución del flujo, cuando pueda resultar inadecuado en relación al funcionamiento correcto del programa. La implantación de esta característica es debido a que existían problemas de constantes errores o «bugs» a la hora de realizar operaciones continuas, por ejemplo: después de configurar una operación cualquiera de dos cantidades, si el usuario teclea erróneamente un número, el sistema aplica esa unidad tecleada al resultado, generando posibles resultados erróneos o, en el peor de los casos bucles infinitos. El sistema de bloqueo se puede definir en: nada estricto (valor 0), poco estricto (valor 1) y estricto (valor mayor o igual a 2).

- **Teclas numéricas (Figura 6.9):** el código implementado es realmente el mismo para todas las teclas numéricas, introduciendo el valor tecleado a un array predefinido. Antes de introducir la unidad en el array, el sistema verifica previamente si no está bloqueado por alguna operación anteriormente hecha (se desbloquea aplicando los limpiadores de registro). A su vez también analiza si el número de dígitos introducidos no superan a lo establecido, permitiendo un máximo de nueve dígitos. Si cumple esta condición, el

dígito es remplazado a una posición del array (EW_9). Fue necesario agregar los dígitos desde la posición final del array, para así representar fácilmente la cantidad en los registros de la calculadora. Continuando la implementación, una vez introducido el dígito en el array un mensaje es enviado para representar visualmente la unidad tecleada en el registro configurador. Poco después del envío del mensaje, el sistema nuevamente verifica el acoplamiento y el número de dígitos introducidos para agregar unidades tecleadas al array (MW_9). Este array es implementado para ser escaneado y utilizado en operaciones de multiplicación automática.

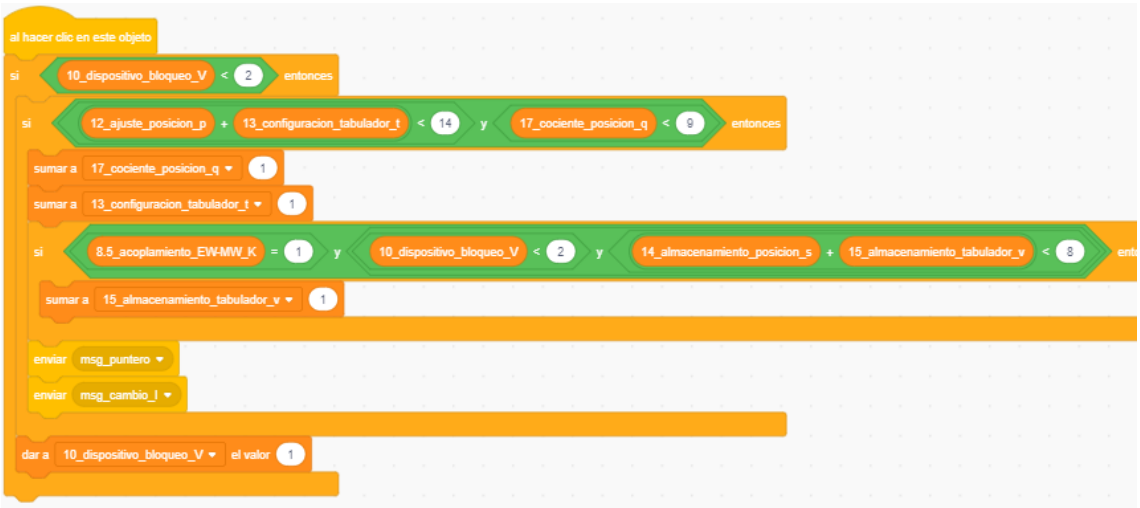

Figura 6.10: Fragmento de código para el cambio de carro izquierdo. Después de verificar si no está bloqueado por alguna operación y cumple con la cantidad de tabulaciones introducida, el sistema incrementa los valores establecidos. Si el bloque fuera para un cambio de carro derecho, los valores se decrementarían.

- **Cambio de carro y Tabulador (Figura 6.10):** el simulador cuenta con dos teclas para el cambio de carro (izquierdo y derecho) y una tecla para la tabulación total izquierda. Estas teclas obligan al puntero a seleccionar una posición específica del array para después operar el dígito de esa posición. La implementación comienza con el sistema de bloqueo, comprobando si es posible continuar con esta funcionalidad. A continuación verifica el número de veces que se ha realizado el cambio de carro hacia una única dirección (izquierda), permitiendo un máximo de trece veces para el registro producto y nueve veces para el registro multiplicador. Poco después de realizar el incremento a las variables *17_cociente_posicion_q* y *13_configuracion_tabulador_t*, nuevamente se comprueba: el acomplamiento de los vectores EW y MW, el bloqueo del dispositivo y el número de veces que se realizó el cambio de carro para asignar una posición, que después será utilizada para las multiplicaciones y divisiones automáticas. Una vez hecha la condición descrita, el sistema envía dos mensajes: el primero para el desplazamiento de los dos punteros y el segundo para la animación del registro configurador. Finalmente, antes de terminar el bloque, el sistema de bloqueo se asigna al estado poco estricta (valor 1).

Figura 6.11: Fragmento de código para la limpieza del «Registro configurador». De todos los limpiadores este es el más completo, aplicándose limpieza a los dígitos de los arrays y asignación de valores de inicio a las variables predefinidas.

- **Limpiadores de registros (Figura 6.11):** ambos simuladores tienen tres objetos predefinidos para limpiar los registros del simulador; visualmente son diferentes ya que en el modelo *C1-13* son palancas y en el modelo *CA1-13* son teclas. Sin embargo, la implementación de cada uno de estos objetos es similar. Empezamos con la implementación del «Registro configurador»: en el código se observa un bucle que recorre el array MW_9, asignando a todas las posiciones el valor 0. Después se le asigna valores de inicio a las variables basadas en el acomplamiento de arrays y contador de posiciones. Seguidamente volvemos a un bucle, recorriendo en este caso el array EW_9 y asignando el valor 0 en todas sus posiciones, para finalmente aplicar valores de inicio a las posiciones de tabulación y sistema de bloqueo. Además, envía dos mensajes para restablecer la posición de los punteros y la animación del registro configurador.

Los limpiadores del «Registro multiplicador» y «Registro producto» tienen la misma implementación. El código se basa en un recorrido de un array, en donde el bucle introducido tiene la misma estructura que la del limpiador del «Registro configurador». La diferencia de estos dos limpiadores se encuentra en el array, es decir, para limpiar el «Registro producto» es necesario asignar el valor 0 a todas las posiciones del array RW_13 y el «Registro multiplicador» al array QW_8.

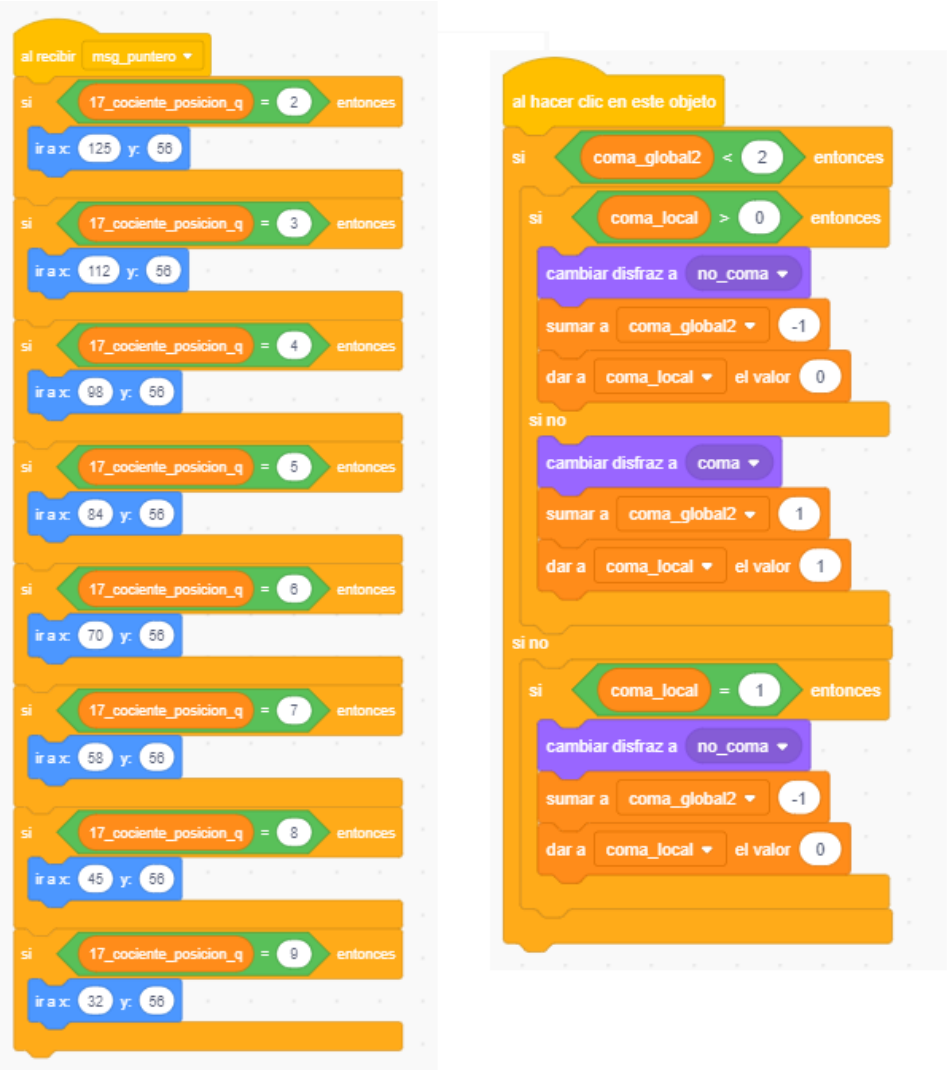

Figura 6.12: Bloque de código para el puntero (izquierda) y coma deslizable (derecha). El bloque derecho realiza un desplazamiento del puntero dentro del escenario; en cambio, el bloque derecho realiza dos tipos de animaciones en función de la colocación de la coma.

- **Punteros (Figura 6.12):** la representación es cuando se realiza un cambio de carro es el puntero. Este objeto implementado de la forma más sencilla posible, en este caso mediante el desplazamiento del objeto a través del escenario. En el bloque izquierdo podemos observar que el desplazamiento va en función de la posición de la variable *17_cociente_posicion_q* para el puntero del «Registro multiplicador». Finalmente la implementación en relación con el puntero del «Registro producto» es de igual forma que el descrito anteriormente.

- **Comas deslizables (Figura 6.12):** la representación de números decimales es primordial, por ese motivo la calculadora mecánica representa una coma deslizable de color blanco-gris. En este caso el diseño del comportamiento se basa en seleccionar el objeto y colocarlo en el lugar querido por el usuario, es decir, dar clic al objeto, desapareciendo poco después y luego hacer nuevamente clic en el lugar deseado dentro del rango posible de la imagen de la regleta. El código muestra dos análisis de la variable coma (*coma_global* y *coma_local*), analizando ambos casos para la selección y aplicación del objeto en el nuevo lugar.

Para finalizar esta sección vamos a resumir de forma breve este apartado. Hemos visto todas la partes del código con comportamiento similar para los dos proyectos, abarcando a su vez el análisis, el diseño visual, la navegación, la interacción, la lógica y los objetos comunes, mostrando la implementación de las partes mas importantes. De tal manera que para los siguientes dos apartados, se expondrán el diseño e implementación de las partes más específicas junto de los dos modelos *C1-13* y *CA1-13*.

6.2 Modelo C1-13

6.2.1. Diseño de la funcionalidad

Antes de realizar el diseño para el modelo manual de diez tecla (*C1-13*), hubo que investigar detenidamente su funcionamiento mecánico y comportamiento operacional a través de vídeos. Sin embargo, existen dos simuladores que sirvieron de gran ayuda para el entendimiento de este tipo de calculadoras[3]. Aunque los simuladores encontrados no pertenecen a este modelo, si que sirvió para comprender su funcionamiento general con respecto a los manuales Facit estudiados.

Después de investigar el funcionamiento de este modelo concreto, pasamos a materializar la idea mediante un diagrama de máquina de estados. El objetivo principal de este diagrama es ayudar al lector a entender los distintos flujos que realiza el programa cuando el usuario encamina una operación en particular. El diagrama diseñado en Bizagi Modeler[4], ofrece de forma visual el flujo que tendría al seguir un camino u otro. Por este motivo vamos a describir este flujo una vez ejecutado el programa para las distintas operaciones programadas (ver Figura 6.13):

- **Suma:** Para que el sistema resuelva una suma tiene que seguir el siguiente flujo:

  1. Empezamos en «Stand-by» o «Modo espera».
  2. El usuario introduce la cantidad en el registro configurador con el estado «Añadir cantidad registro configurador», y volvemos a «Stand-by».

[3]Simuladores encontrados en: http://www.rechenautomat.de/Facit/Rechner.php
[4]Aplicación descargada en: https://www.bizagi.com/es/plataforma/modeler

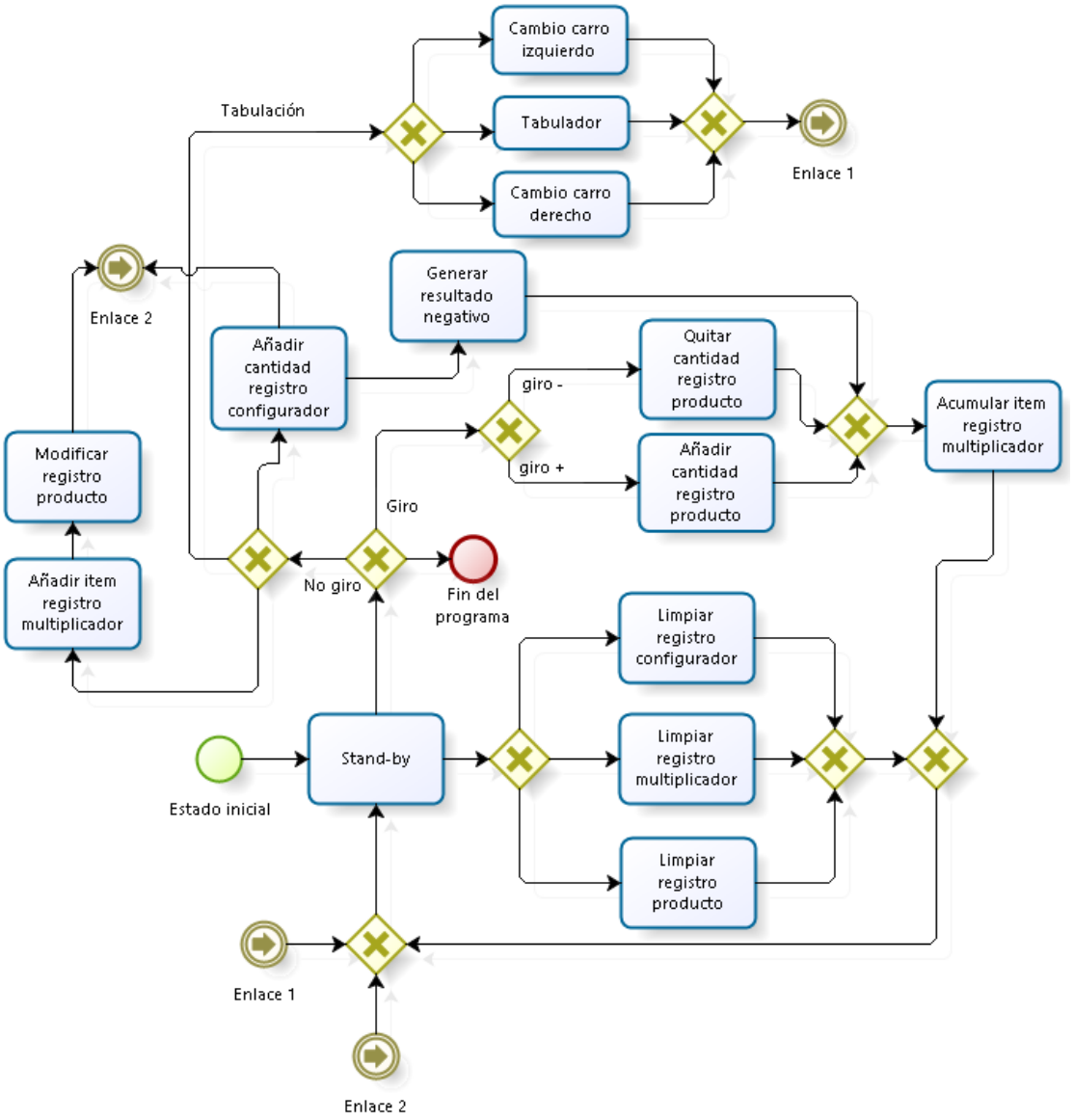

Figura 6.13: Diagrama de máquina de estados del funcionamiento de la calculadora Facit *C1-13*. Cada camino realiza una operación o funcionalidad distinta en el programa.

3. En este caso, como se va a realizar una suma, redirigimos el flujo al estado «Añadir cantidad registro producto» seleccionando la opción giro positivo.
4. Después de introducir la cantidad, pasamos al estado «Acumular item registro multiplicador», añadiendo el número de veces que introducimos la misma cantidad al registro producto. Seguidamente volvemos al estado «Stand-by».
5. Finalmente borramos el registro configurador con el estado «Limpiar registro configurador», para después volver nuevamente al estado «Stand-by».
6. Volvemos a repetir los pasos 1 al 5 para sumar mas cantidades en el sistema.

- **Resta:** El flujo del programa para resolver una resta es el siguiente:

1. Realizamos una vez el flujo Sumar, aplicando la primera cantidad.
2. Después de tener la cantidad a la que vamos a restar, empezamos introduciendo la cantidad en el registro configurador con el estado «Añadir cantidad registro configurador».
3. En este caso al ser una resta, redirigimos el flujo al estado «Quitar cantidad registro producto» seleccionando la opción giro negativo. Después de quitar la cantidad, pasamos al estado «Acumular item registro multiplicador», añadiendo de forma negativa el numero de veces que introducimos la misma cantidad al registro producto.
4. Finalmente borramos el registro configurador con el estado «Limpiar registro configurador», para después volver nuevamente al estado «Stand-by».
5. **Opcional:** Si realizamos una resta negativa el resultado marcado no es correcto, por esa razón es primordial aplicar el estado «Generar resultado negativo», previamente añadido la cantidad al registro configurador con «Añadir cantidad registro configurador».

- **Multiplicación:** Para que el sistema resuelva una multiplicación tiene que seguir el siguiente flujo:

  1. Empezamos en «Stand-by» o «Modo espera».
  2. El usuario introduce el multiplicando en el registro configurador con el estado «Añadir cantidad registro configurador», poco después el sistema regresa al estado «Stand-by».
  3. A continuación agregamos el multiplicador al registro multiplicador con el estado «Añadir item registro multiplicador» para poco después generar un resultado gracias al estado «Modificar registro producto».
  4. Si el multiplicador es largo, es necesario aplicar tabulaciones para introducir las centenas, millares, etc. Por eso es necesario desplazar el puntero hacia la izquierda con «Cambio carro izquierdo».
  5. Después de realizar el desplazamiento del puntero aplicamos nuevamente el paso 3.
  6. Finalmente, el registro producto lanza el resultado gracias al estado «Modificar registro producto» explicado en el paso 3.

- **División:** El flujo del programa para resolver una división es el siguiente:

  1. Después de estar en «Stand-by», introducimos el dividendo con «Añadir cantidad registro configurador».
  2. Realizamos una tabulación total a la izquierda con «Tabulador» y volvemos a «Stand-by».
  3. Redirigimos el flujo nuevamente seleccionando giro positivo aplicando los estados «Añadir cantidad registro producto», «Acumular item registro multiplicador».
  4. Borramos los registros multiplicador y configurador con los estados «Limpiar registro configurador» y «Limpiar registro multiplicador», para después repetir los pasos 1 y 2.

5. A continuación redirigimos el flujo a «Quitar cantidad registro producto», restando el registro producto hasta que no pueda restarse más. A su vez, el flujo se desplaza por «Acumular item registro multiplicador» añadiendo la respuesta de la división para finalmente volver a «Standby»
6. Desplazamos el puntero hacia la derecha con «Cambio carro derecho».
7. Repetimos el paso 5 hasta no poder restar más el registro producto.

6.2.2. Componentes específicos

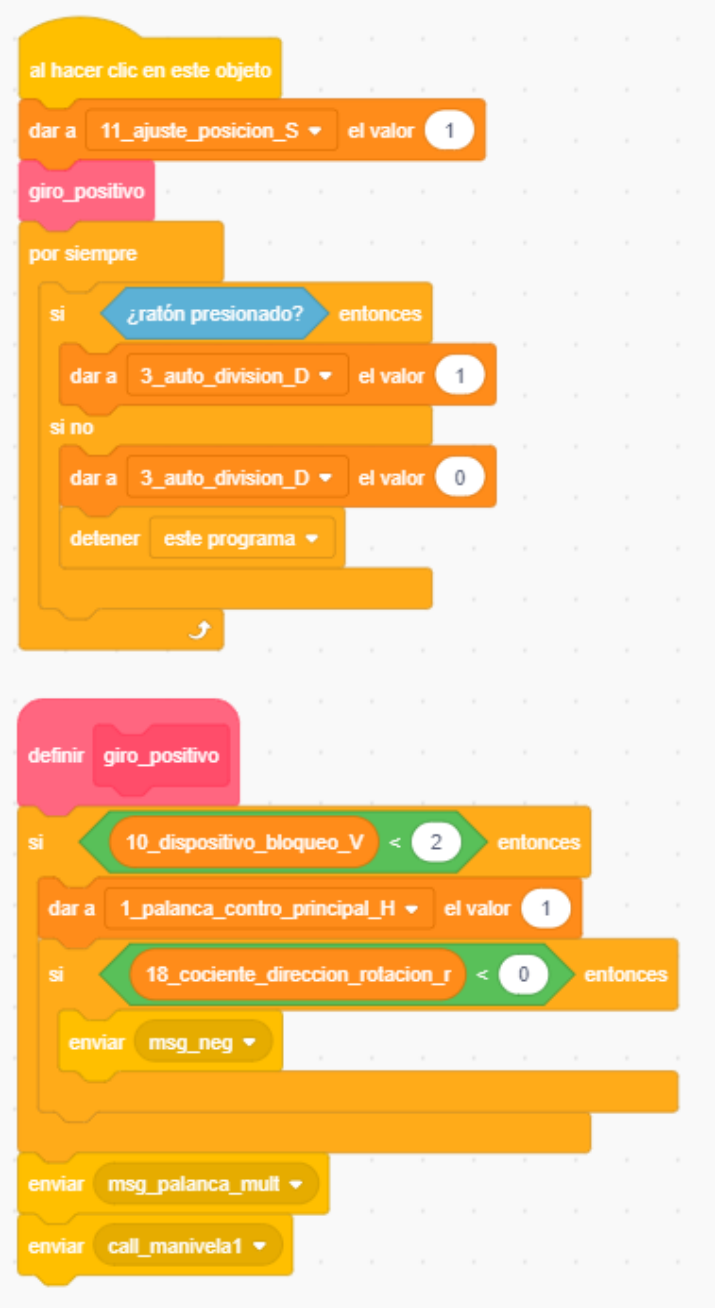

Figura 6.14: Bloques de código utilizados para realizar un giro positivo de manivela en el modelo *C1-13*. El código es igual para el giro negativo.

Una vez planeada y diseñada la idea, ahora pasamos a las partes más específica para el modelo *C1-13*. En esta sección vamos a detallar las partes principales que diferencian este modelo de la versión *CA1-13*, explicando de forma breve el comportamiento ligado a su implementación.

Como se describió anteriormente, el modelo *C1-13* en diseño visual es casi igual que el modelo *CA1-13*, diferenciándose únicamente en el número de teclas y palancas. Además su funcionalidad es mucho menor que el modelo *CA1-13*, pero sigue cumpliendo con las funcionalidades básicas para realizar operaciones aritméticas principales.

- **Giro positivo y negativo de manivela (Figura 6.14):** En este caso tenemos la implementación utilizada para realizar un giro positivo cuando es presionado el objeto. El primer bloque realiza una asignación a la variable *11_ajuste_posicion_S* (esta variable es determinante a la hora de realizar una operación) para después llamar al bloque *giro_positivo* y comprobar si el simulador no está bloqueado por alguna operación antes hecha, evitando inconsistencias a la hora de realizar cálculos. Además verifica si el giro anterior es negativo, enviando un mensaje para cambiar el color del visualizador del registro multiplicador y finalmente envía dos mensajes para las animaciones del objeto manivela. Después de terminar el bloque *giro_positivo*, éste continua con un bucle infinito en donde asigna un valor a la variable *3_auto_division_D* (1 si esta presionado y 0 en caso contrario) activando así la opción operadora del simulador. La implementación para el giro negativo sigue el mismo formato, diferenciándose en la asignación de la variable *11_ajuste_posicion_S*, aplicándose el valor 0.

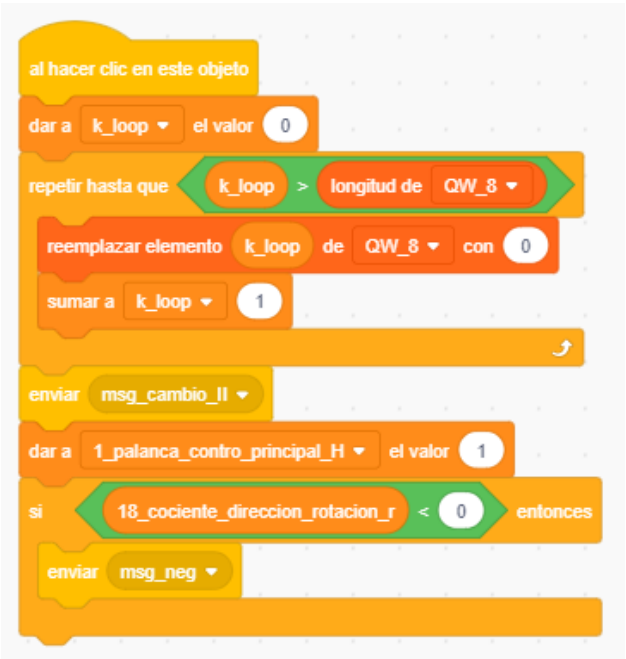

Figura 6.15: Bloque de código utilizado para limpiar el registro multiplicador en el modelo *C1-13*. Asigna a cada posición del array el valor 0.

- **Limpiador del registro multiplicador (Figura 6.15):** como se describió anteriormente, este componente es muy similar al modelo *CA1-13*. Sin embargo, este modelo tiene una peculiaridad especial para que el funcionamiento de las operaciones sea semiautomático. Como se puede observar en la implementación, después de limpiar el array QW_8 el sistema envía un mensaje para cambiar el disfraz del registro multiplicador. Además, asigna un valor para el tipo de operación en función del giro de manivela que se realice. Finalmente la última condición cambia el color de la señal de la dirección de rotación poniéndola en modo normal (color negro).

6.3 Modelo CA1-13

6.3.1. Diseño de la funcionalidad

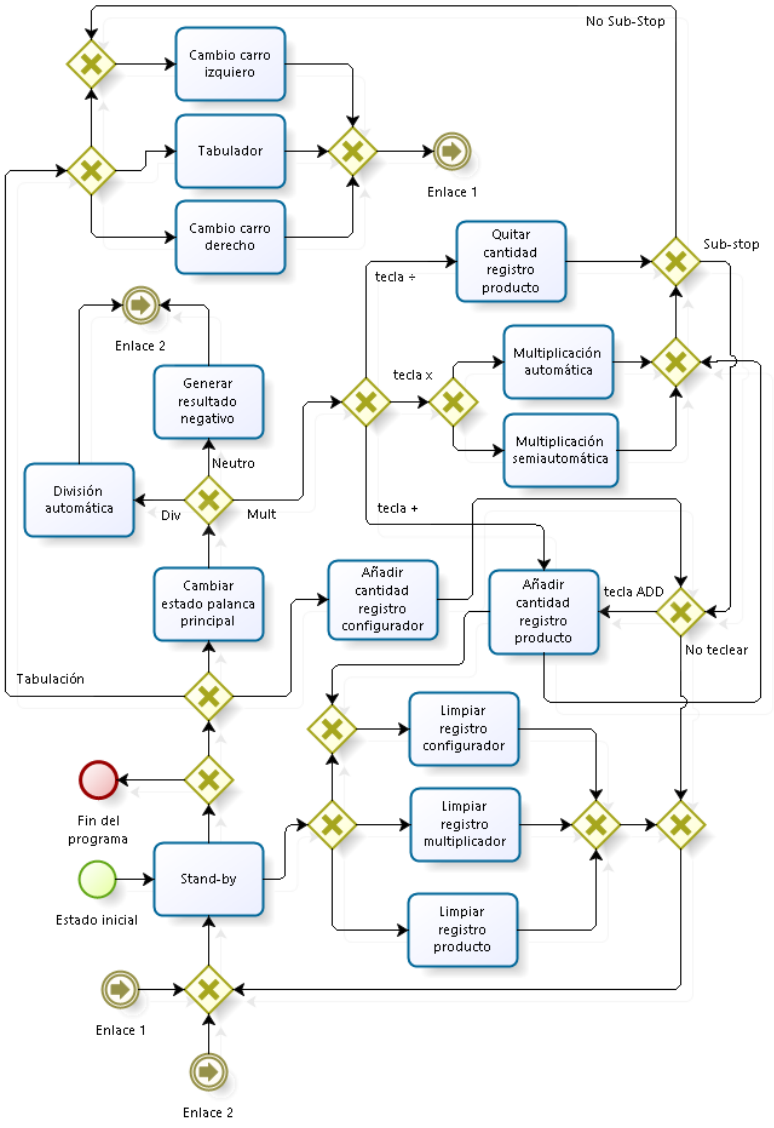

Figura 6.16: Diagrama de máquina de estados para el modelo *CA1-13*.

Una vez explicado el modelo *C1-13*, ahora pasamos a detallar el segundo proyecto desarrollado. En este caso se trata del modelo *CA1-13*. Este tipo de calculadora, visto en capítulos anteriores, se describe como un modelo motorizado y ofrece muchas mas funcionalidades a la hora de desarrollar operaciones aritméticas. Por ese motivo, hubo que investigar detenidamente el funcionamiento mecánico y operacional al igual que el modelo *C1-13*. Pero la diferencia está en que resultó mucho más sencillo al utilizar un simulador[5] para este modelo, ayudando a entender el funcionamiento general con apoyo de los manuales Facit estudiados.

Una vez investigado el funcionamiento para este modelo, pasamos nuevamente a materializar la idea a través de un diagrama de máquina de estados, ofreciendo al lector el diseño desarrollado en Bizagi Modeler antes de ser implementado. El diagrama de máquina de estados ofrece una visión del flujo que tiene el programa, una vez iniciado el «Estado inicial». A continuación describiremos el flujo para cada una de las operaciones que se pueden realizar con el simulador (ver Figura 6.16):

- **Suma:** Para que la calculadora resuelva una suma, el sistema tiene que seguir el siguiente flujo:
  1. Empezamos en «Stand-by» o «Modo espera».
  2. En este caso podemos elegir dos flujos que realizan la misma operación:
     a) Nos movemos al primer estado «Añadir cantidad registro configurador», para después redirigir el flujo hacia «Añadir cantidad registro producto» al seleccionar tecla ADD, y finalmente «Limpia registro configurador» para después volver al estado «Stand-by».
     b) Seleccionar «Añadir cantidad registro configurador» volviendo después al estado «Stand-by». A continuación vamos al estado «Cambiar estado de palanca principal», redirigiendo el flujo hacia *Mult* para después elegir tecla +, seleccionando el estado «Añadir cantidad registro producto», para después realizar el estado «Cambio carro izquierdo», volviendo posteriormente a «Stand-by».
     Si seleccionamos este flujo, es necesario aplicar el estado «Cambio de carro derecho» y «Limpiar registro configurador».
  3. Volvemos a repetir los pasos anteriores para sumar más cantidades.

- **Resta:** El flujo del programa para resolver una resta es el siguiente:
  1. Realizamos una vez el flujo Sumar, aplicando la primera cantidad.
  2. Después de tener la cantidad a la que vamos a restar, empezamos introduciendo la segunda cantidad en el registro configurador con el estado «Añadir cantidad registro configurador», volviendo poco después al estado «Stand-by».
  3. Redirigimos el flujo al estado «Cambiar estado palanca principal», para después elegir la tecla ÷ y seleccionar el estado «Quitar cantidad registro producto». Después de quitar la cantidad, tenemos dos flujos:

[5]Simulador Facit CA1-13 ubicado en: www.rechenautomat.de/Facit/Sim-1/CA1-13.html

a) **Aplicar Sub-stop:** Si elegimos este flujo, el sistema se dirige al estado «Stand-by».

b) **No aplicar Sub-stop:** Si elegimos este flujo, el sistema se dirige al estado «Cambio carro izquierdo», terminando después en «Stand-by».

4. **Opcional:** Si realizamos una resta negativa el resultado marcado no es correcto, por esa razón es primordial aplicar el estado «Generar resultado negativo», seleccionando la opción *Neutro* en «Cambiar estado palanca principal».

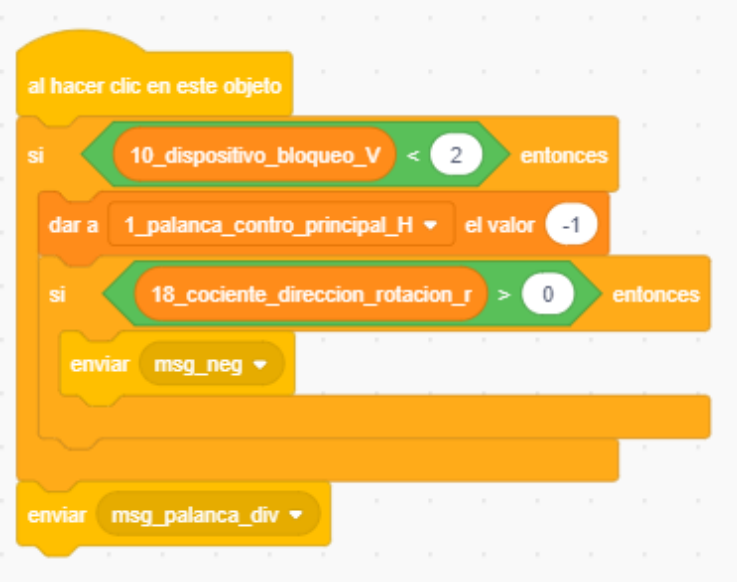

Figura 6.17: Código de la palanca de control deslizable. En este caso el estado de la palanca es «Div», debido a que la variable asignada es -1.

- **Multiplicación:** Para que resuelva la calculadora una multiplicación, el sistema tiene que seguir el siguiente flujo:

  1. Empezamos en «Stand-by» o «Modo espera».
  2. El usuario redirige el flujo al estado «Cambiar estado palanca principal» para elegir la opción *Mult*. En este caso tenemos dos posibles estados a escoger, por un lado «Multiplicación automática» y por otro «Multiplicación semiautomática». La multiplicación semiautomática es muy parecida al modelo *C1-13* pero agregando las cantidades al registro multiplicador con la tecla +.
  3. Si elegimos la multiplicación automática, tenemos que seleccionar primeramente el estado «Añadir cantidad registro configurador» para después aplicar el estado «Multiplicación automática».

- **División:** El flujo del programa para resolver una división es el siguiente:

  1. Después de estar en «Stand-by», introducimos el dividendo con «Añadir cantidad registro configurador», volviendo poco después a «Stand-by».

2. Realizamos una tabulación total a la izquierda con «Tabulador» y volvemos a «Stand-by».
3. Redirigimos el flujo nuevamente seleccionando el estado «Añadir cantidad registro configurador» para después seleccionar «Añadir cantidad registro producto».
4. Volvemos añadir el divisor con «Añadir cantidad registro configurador» y después «Tabulador», seleccionando a continuación el estado «Cambiar estado palanca principal» para después redirigir el flujo hacia el estado «División automática» a través de la opción *Div* y volver finalmente a «Stand-by».

6.3.2. Componentes específicos

Luego de plantear y diseñar la idea, ahora pasamos a las partes más específicas para el modelo CA1-13. Para finalizar esta sección vamos a describir las partes más importantes que diferencian este modelo con la versión anterior *C1-13*, detallando de forma resumida el comportamiento que tiene cada parte.

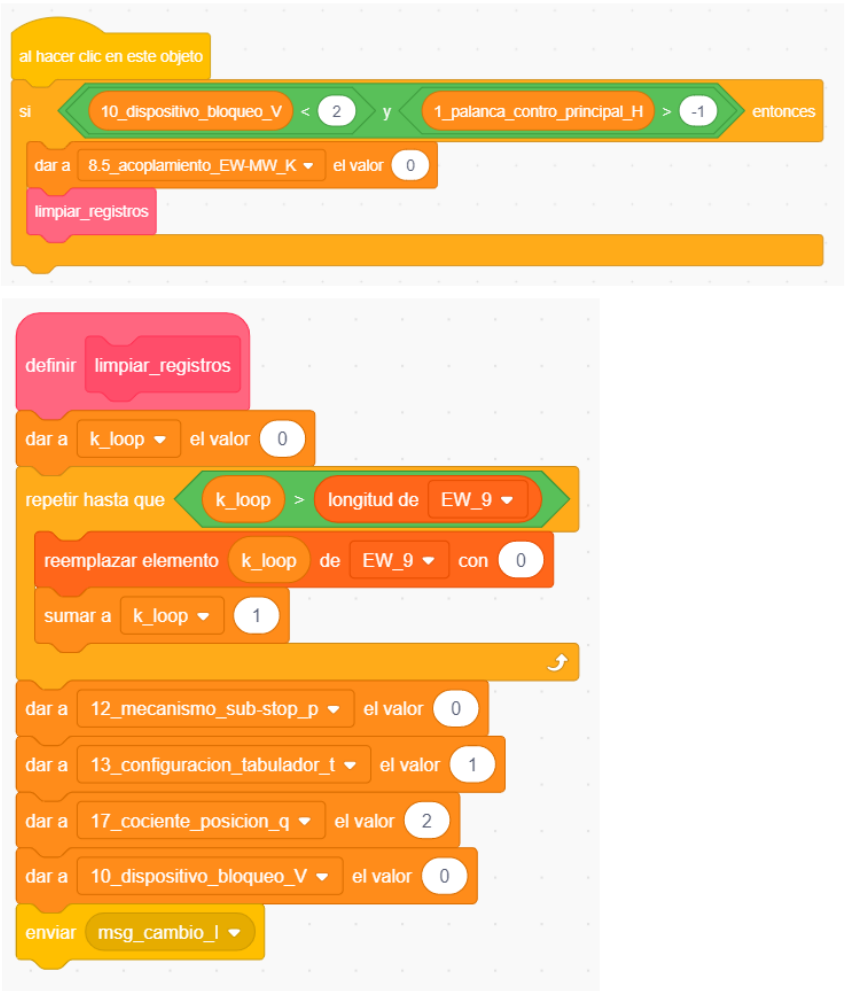

Figura 6.18: Código de la tecla X para el modelo *CA1-13*. Este conjunto de bloques genera una preparación de la multiplicación automática.

La calculadora modelo *CA1-13* tiene un diseño visual parecido al *C1-13*, a diferencia de que las funcionalidades que lleva el *CA1-13* son más numerosas que las de su hermana *C1-13*. Por ese motivo vamos a mostrar cada una de esas componentes extras:

- **Palanca de control deslizante (Figura 6.17):** este componente es utilizado para realizar las distintas operaciones aritméticas introducidas en el simulador. La implementación comienza con el sistema de bloqueo, analizando si no está bloqueado por alguna operación antes hecha (analiza si es un bloqueo estricto). Después asigna a la variable *1_palanca_contro_principal_H* un valor predefinido para que realice los tipos de operaciones basados en giros de rotación mecánica (1 es *Mult*, 0 es *Neutro* y -1 es *Div*). Después de asignar la variable, el sistema analiza si la dirección del giro es negativa, para después enviar un mensaje al objeto y cambie el color de la señal de la dirección de rotación. Finalmente el último mensaje realiza una animación visual a la palanca de control.

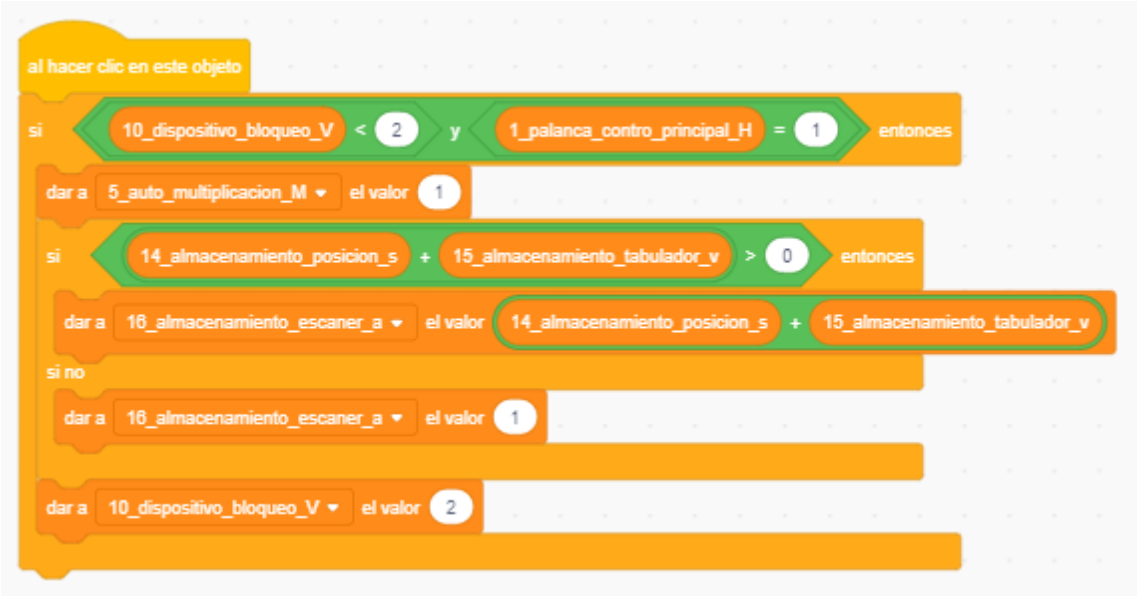

Figura 6.19: Código de la tecla igual. Para ejecutar la multiplicación automática la variable *5_auto_multiplicacion_M* es la encargada de dar comienzo al proceso. Además la verificación del las variables de almacenamiento elimina errores de asignación de variables durante el acoplamiento de números.

- **Tecla X (Figura 6.18):** Para la tecla X, su implementación abarca una configuración especial a la hora de asignar las variables. Al comenzar el bloque de código volvemos a tener el sistema de bloqueo, verificando si el sistema no está bloqueado por alguna operación antes hecha. Poco después modificamos la variable de acoplamiento antes de preparar el multiplicando en el sistema. Poco después, la tecla de forma automática limpia el «Registro configurador» con la llamada a la función *limpiar_registros*, asignando en cada posición el número cero al array (EW_9) para luego reconfigurar las variables, que serán utilizadas para introducir nuevamente otro valor (multiplicador) a la operación.

- **Tecla igual (Figura 6.19):** La tecla igual es utilizada únicamente para realizar multiplicaciones de forma automática. El bloque de código comienza con la verificación de bloqueo (analiza si no esta bloqueado por otra operación) y si la palanca de control principal esta activada (posición *Mult*). Después se asigna un valor a la variable *5_auto_multiplicacion_M* para que de comienzo a la multiplicación automática, pero antes de ejecutar el proceso, es necesario verificar las posiciones del almacenamiento de los dígitos que se van a multiplicar. Por ese motivo, es importante utilizar el número de cantidades introducidas para realizar el escaneo de los números en el sistema de acoplamiento.

- **Teclas ADD, ÷ y + (Figura 6.20):** La tecla ADD se utiliza para agregar de forma rápida cantidades una tras de otra (suma rápida). Por ese motivo, al igual que la tecla +, su implementación se basa en el tipo de giro mecánico positivo. Sin embargo, tiene una diferencia con la tecla + y es la activación de la variable ligada al mecanismo Substop (mecanismo que evita el cambio de carro automático). Como se puede ver en la implementación, el bloque comienza con el sistema de bloqueo, para después asignar el valor de comienzo de rotación a la variable *9_sentido_rotacion_rotor_U*, dando al sistema de bloqueo el valor uno (poco estricto) y activar el mecanismo SUBSTOP con un uno a la variable *11_mec_substop_S*. A su vez se analiza el tipo de rotación realizado por el mecanismo rotación cociente, cambiándola a cero si el valor es negativo (negativo significa que se utilizó para divisiones). Finalmente la tecla ÷ es igual que la tecla + a diferencia de que el valor para el tipo de rotación es negativo, asignando a la variable un -1.

Figura 6.20: Código de la tecla ADD. La variable *9_sentido_rotacion_rotor_U* es la encargada de iniciar la operación en la lógica y la variable *18_cociente_direccion_rotacion_r* es la encargada de asignar un tipo de giro para realizar una operación determinada.

- **Tecla NEG y SUB-STOP (Figura 6.21):** La tecla NEG se emplea para asignar de forma rápida un tipo de giro mecánico. Existen dos tipos de giros: positivo, utilizado para sumar, restar y multiplicar; en cambio, el giro negativo es usado para dividir. Como se puede observar en el primer bloque, su implementación comienza con el sistema de bloqueo, para después asignar el tipo de giro negativo a la variable *18_cociente_direccion_rotacion_r* (multiplica por -1) y a su vez realiza un cambio de color a la señal de dirección de rotación. Finalmente el último bloque es para la tecla SUB-STOP: en este caso se asigna un valor de uno a fin de activar esa opción.

Figura 6.21: Código de las teclas NEG y SUB-STOP. La tecla NEG utiliza el primer bloque, asignandose el tipo de rotación a la variable *18_cociente_direccion_rotacion_r* un -1 (vuelta negativa). La tecla SUB-STOP tiene implementado el segundo bloque, donde se desactiva el cambio de carro automático al asignar un 1.

CAPÍTULO 7
Divulgación patrimonial

En este capítulo se muestra la historia, objetivos y compromiso que tiene el Museo de Informática de la Escola Tècnica Superior d'Enginyeria Informàtica con la sociedad. Además de describir lo que contribuye el presente trabajo al museo.

7.1 Museo de Informática

El museo (ver Figura 7.1) se inauguró oficialmente el 11 de diciembre de 2001, y ha ido mejorando gracias al apoyo de profesores, estudiantes y entidades tanto públicas como privadas. Por ese motivo, el 13 de mayo de 2013 el Museo de Informática fue reconocido oficialmente por la Conselleria d'Educació, Cultura i Esport como museo oficial de la Comunitat Valenciana[1], resolución que se hizo pública en el DOCV, con fecha de 28 de mayo de 2013 [7].

Figura 7.1: Logo del Museo de Informática de la Universitat Politècnica de València.

Sus fondos públicos o privados han llevado a la continua mejora del museo, siendo utilizadas en la realización de importantes eventos tales como la exposición 40 años de Informática celebrada el año 2006, el evento anual «Retrópolis» divulgando, promocionando y preservando la cultura del videojuego y de la informática clásica. Además, entre sus principales objetivos de la entidad, esta la organización de diversas actividades culturales, como conferencias, exposiciones temporales y certámenes de monografías sobre la historia, evolución y futuro de la Informática.

[1]Para acceder al DOCV: http://museo.inf.upv.es/wp-content/uploads/2012/11/Reconocimiento-oficial-Museo-de-Informática.pdf

Finalmente, dentro de los diecisiete objetivos para el desarrollo sostenible puesta por Naciones Unidas (ODS)[2], el presente trabajo cumple los objetivos cuatro (Educación de calidad) y diez (Reducción de las desigualdades), gracias a la contribución que se realiza a la difusión patrimonial. Además cabe destacar que el Museo de Informática está integrado dentro del Consejo Internacional de Museos (ICOM, International Council of Museums) desde el año 2015 [7].

7.2 Web del Museo

En la Figura 7.2 podemos ver la página principal de la web del Museo de Informática[3], donde se puede encontrar muchos tipos de contenidos destacados. Una vez dentro de la web, tenemos la posibilidad de acceder a muchas páginas muy interesantes relacionadas con la informática, teniendo las exposiciones, información sobre todos los eventos que se realizan, etc.

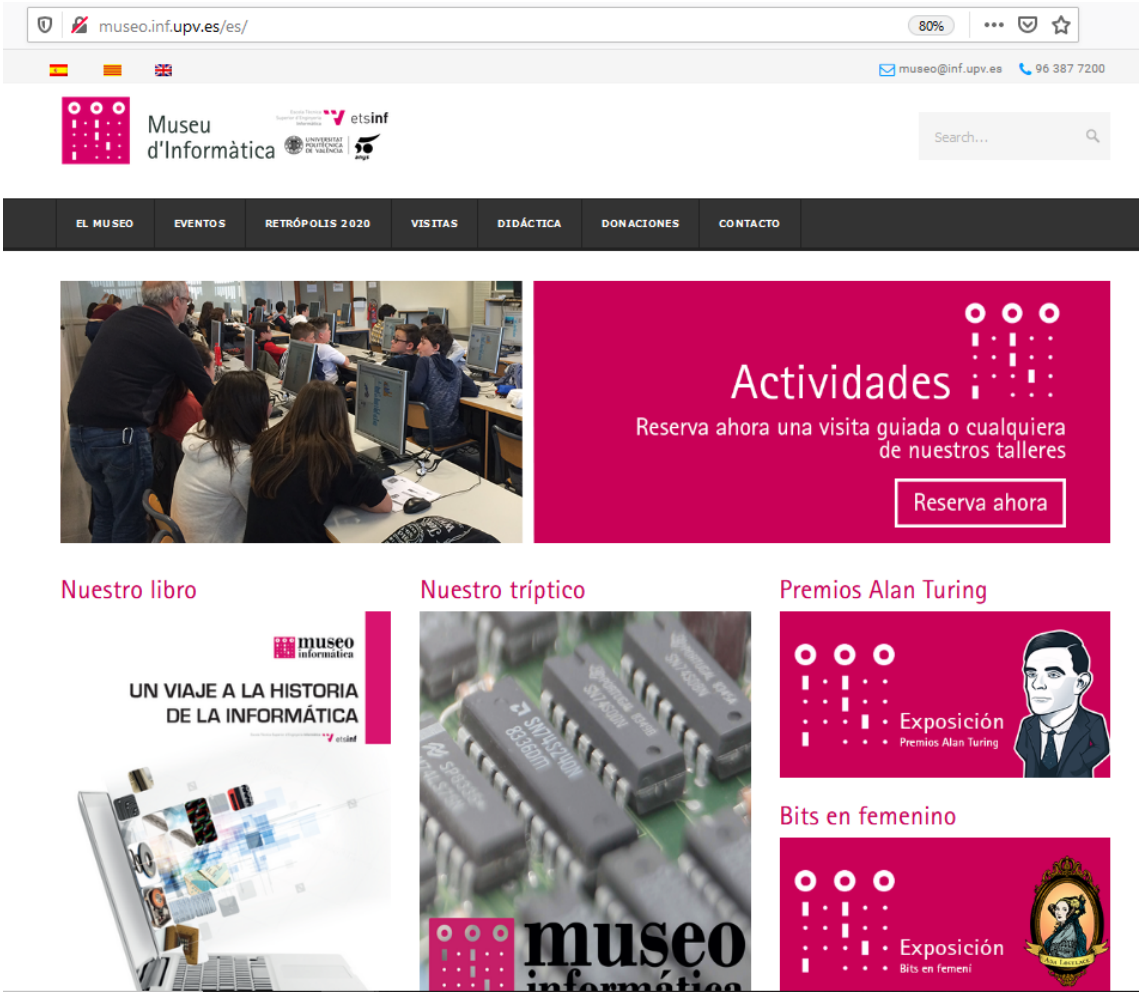

Figura 7.2: Página principal del Museo de Informática de la Universitat Politècnica de València. Desarrollado en Wordpress un CMS que ayuda al Webmaster a mejorar la gestión del contenido principal de la web.

[2]Informe de los Objetivos de Desarrollo Sostenible: https://unstats.un.org/sdgs/report/2019/The-Sustainable-Development-Goals-Report-2019_Spanish.pdf

[3]Para visitar la web del Museo de Informática: http://museo.inf.upv.es/es/

Además de mostrar artículos divulgativos didácticos donde abarcan la historia, la evolución y los proyectos desarrollados en Scratch sobre un tema en concreto basados en la retro-informática. Como ya se describió anteriormente, el objetivo principal del trabajo es la divulgación de la historia mediante el desarrollo de proyectos interactivos en Scratch, enseñando tanto la historia del proyecto, mostrando el funcionamiento de las calculadoras Facit de mediados del siglo XX, y además enseñar de forma indirecta el «Pensamiento Computacional» a través de la implementación desarrollada.

Gracias al continuo trabajo que realiza la Escola Tècnica Superior d'Enginyeria Informàtica, se puede enseñar a programar a personas iniciadas en el tema. Jóvenes y adolescente pueden encontrar un repositorio amplio de proyectos realizados por alumnos de la Universidad (incluyendo el presente trabajo), enseñando herramientas que acercan al desarrollo e implementación de aplicaciones básicas, teniendo como ejemplo el lenguaje de programación Scratch. Todos los proyectos planteados, programados y mostrados en la web del museo, relaciona las matemáticas, la informática y la historia relacionada a cada proyecto. Entre los proyectos podemos encontrar desde simuladores de calculadoras antiguas hasta juegos clásicos muy famosos.

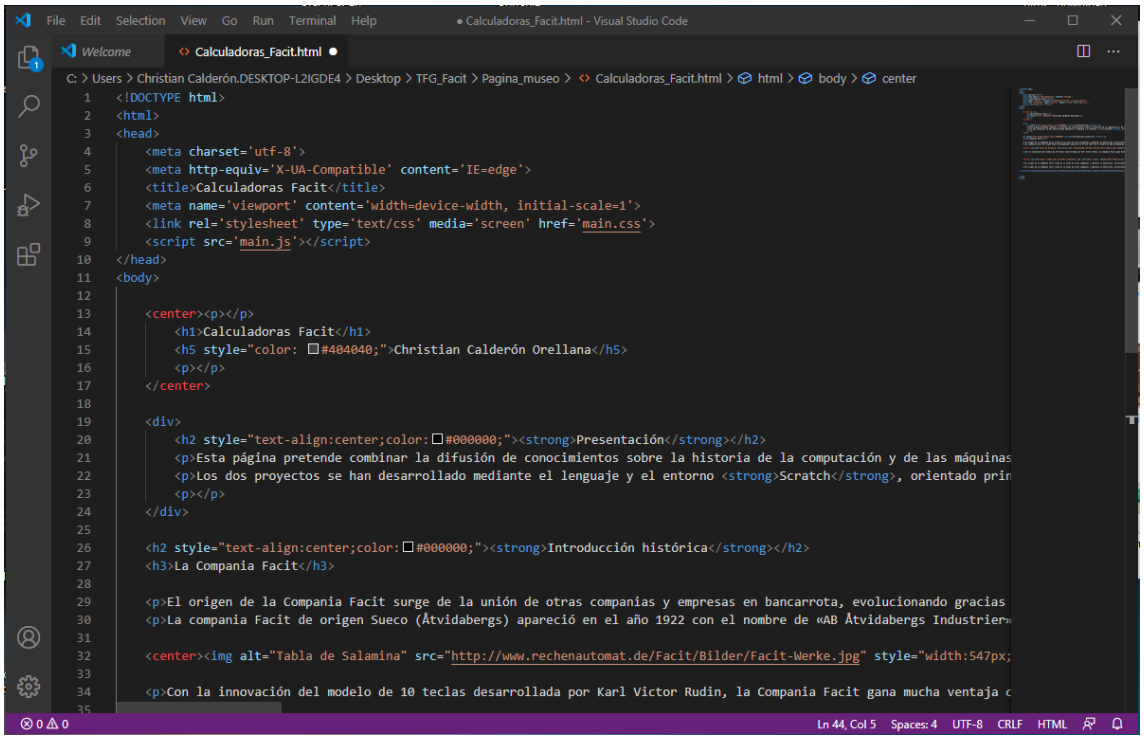

Figura 7.3: Código HTML de la página desarrollada. En este caso se utiliza el editor de código fuente Visual Studio Code tanto para la estructura como el estilo.

La pagina web tiene un diseño sencillo e intuitivo, debido a que su desarrollo e implementación está ligada a una base principal (Wordpress). Wordpress[4] es un «CMS» o gestor de contenidos que permite administrar, crear y modificar contenidos web de manera fácil y sencilla. Dentro del «CMS» podemos crear páginas que pueden ser modificadas con lenguajes de marcas (HTML) y páginas de esti-

[4]Aplicación web disponible en: https://es.wordpress.org/

los (CSS). Por este motivo, se desarrolla para este caso una página con descripción referente al presente trabajo (Calculadoras Facit).

Como se puede observar en la Figura 7.3 el desarrollo es realizado con el editor de código fuente Visual Studio Code[5], utilizando mayormente *Hypertext Markup Language* (HTML), un lenguaje de marcas muy común para el desarrollo de páginas web. Además de este lenguaje de marcas, también se utiliza Cascading Style Sheets (CSS) para el estilo de la página y JavaScript para la animación básica. Como resultado final tenemos la página web tal y como se observa en la Figura 7.4.

Figura 7.4: Resultado de la página una vez creada. La estructura, modelo y estilo están pensados de que sean iguales al de la página original del Museo de Informática.

Su diseño se basa en la incorporación de pocas secciones y subsecciones, introducidos por etiquetas. Cada sección tiene un párrafo prediseñado, mostrando el texto introducido sobre el tema específico. A su vez se introduce imágenes con referencias externas o internas al servidor y un código embebido para cada uno de los proyectos Scratch realizados.

[5] Aplicación descargada en: https://code.visualstudio.com/

CAPÍTULO 8
Conclusiones

Para finalizar esta memoria el presente capítulo ofrece una visión retrospectiva sobre el trabajo realizado. Por ese motivo, vamos analizar y evaluar cada uno de los objetivos planteados al principio del mismo, al igual que una introspección acerca de lo llevado a cabo en el trabajo. Además se proyectará las posibles ideas que serían capaz de expandir para un futuro los proyectos realizados.

8.1 Consideraciones finales

Dentro de los objetivos principales cumplidos para este trabajo es ofrecer datos históricos y técnicos sobre las calculadoras Facit, especialmente los modelos *C1-13* y *CA1-13*. Además, analizar, diseñar e implementar mediante el lenguaje de programación Scratch los modelos explicados.

Dentro del carácter histórico sobre las herramientas de cálculo a lo largo de la historia, hemos visto primeramente una percepción general de la evolución de cada herramienta, desde su origen hasta el fin de la época de oro de las calculadoras mecánicas. Por medio de esto, deseamos que el lector tenga ampliamente una visión crítica y curiosa sobre la evolución de estas herramientas de cálculo. A lo largo del análisis histórico, se han mostrado y detallado las características principales de cada uno de estos instrumentos de cálculo complejos, observando su versatilidad e ingeniosidad.

Después pasamos al origen de una de las compañías más importantes del siglo XX, mostrando un estudio detenido sobre la evolución de la organización a través de su innovación, mediante la introducción de varias mejoras a sus modelos calculadores, como por ejemplo, el sistema que incluye diez teclas al mecanismo Odhner, llamado también "Principio Facit". A partir de ese instante genera un amplio crecimiento en sus diversos modelos, incluyéndose las versiones electromecánicas que realizan operaciones automáticas (*CA1-13*). Además, se ha detallado en forma de tabla las distintas versiones de calculadoras mecánicas desarrolladas por la compañía Facit, mostrando sus principales características y patentes utilizadas para cada modelo. Asimismo, una vez recorrido la historia de la compañía, pasamos a mostrar una visión general sobre el manejo de las calculadoras Facit, describiendo las partes principales, secundarias y manejo a través de pasos sobre los modelos *C1-13* y *CA1-13*.

En el cuarto capítulo cambiamos totalmente el tema, describiendo la historia, la idea y funcionamiento de la plataforma Scratch, empezando con una introducción básica acerca del objetivo principal que ofrece esta herramienta a los jóvenes (enseñanza del «Pensamiento Computacional»). Además se realiza una descripción sobre el entorno y componentes principales que ofrece la herramienta para la creación, modificación y puesta en marcha de proyectos. A pesar de ser una herramienta sumamente simple, es capaz de implementar algoritmos complejos desde una serie de instrucciones básicas. Al ser un lenguaje de alto nivel, resulta difícil implementar funcionalidades a muy bajo nivel. No obstante, ese requisito queda fuera de la finalidad de este trabajo.

Justo después de adquirir los conceptos principales acerca de las calculadoras Facit y la plataforma Scratch, pasamos a cumplir uno de los objetivos esenciales de este trabajo: análisis, diseño e implementación de las calculadoras Facit *C1-13* y *CA1-13*. Cada uno de los proyectos tiene una calculadora virtual que se comporta de manera única, tal y como se ve en los distintos diagramas de máquina de estados, en donde simulan el comportamiento de los modelos Facit descrito anteriormente. También lleva implementado funcionalidades de navegación e interacción con el usuario, tal y como se describe en el diagrama de Casos de uso. De esta forma, tenemos un resultado final adecuado para que el usuario interactúe de forma visual sobre el contexto planteado en el trabajo. Además, se describe de manera detallada el código implementado, prestando atención en las partes mas relevantes que puedan resultar complejas de entender o resultaron difíciles de desarrollar.

Finalmente, se desarrolla una sencilla página web realizada con HTML y CSS sobre el tema del presente trabajo, agregando los dos proyectos realizados y un breve resumen del contexto histórico, que será agregada posteriormente a la web oficial del Museo de Informática.

8.2 Trabajo futuro

Los proyectos pueden ser ampliados en varias formas: primeramente, existe la posibilidad de extender las funcionalidades a partir de las ya construidas, por ejemplo, aumentar más modos de juego a partir de la dificultad y las operaciones realizadas por las calculadoras, generando mayor curiosidad al usuario cuando realiza operaciones aleatorias en el simulador; segundo, generar un historial de los puntos acumulados por el usuario cuando hacen un desafío en concreto, mostrando después una animación dependiendo de los puntos conseguidos; tercero, posibles mejoras en efectos visuales y sonoros al utilizar los simuladores, debido a que no tiene agregado esos detalles.

A parte de las ampliaciones descritas anteriormente, también existe la posibilidad de poder extender la gama de calculadoras en forma de simuladores, debido a que la compañía Facit tiene muchos modelos desarrollados. De esta manera, se puede aportar al Museo de Informática de la Universidad Politécnica de Valencia mayor difusión y popularización sobre este tipo de calculadoras, divulgando de manera virtual este patrimonio histórico al resto del mundo.

Bibliografía

[1] Osmo Pekonen. Gerbert d'Aurillac: Mathematician and Pope. *The Mathematical Intelligencer*, volumen 22, número 4, páginas 67-70, 2000.

[2] Félix Garcia Merayo. Pascal: El científico, el filósofo, el teólogo. *Manual formativo de ACTA*, ISSN 1888-6051, número 39, páginas 49-63, 2006.

[3] J. R. Ratcliff. Samuel Morland and His Calculating Machines c.1666: The Early Career of a Courtier-Inventor in Restoration London. *The British Journal for the History of Science*, volumen 40, número 2, páginas 159-179, 2007.

[4] Svante Kolsgård. VÄRLDSKONCERNEN FACIT OCHINDUSTRIKÖPINGEN ÅTVIDABERG1920-tal – 1970-tal. *Svante Kolsgård är lärare i historia vid Linköpings universitet*, Brukskultur Åtvidaberg, 2002.

[5] Jesús Moreno León, Gregorio Robles, Marcos Román Gonzáles. Dr. Scratch: Análisis Automático de Proyectos Scratch para Evaluar y Fomentar el Pensamiento Computacional. *RED-Revista de Educación a Distancia*, 46(10) 15-Sep-2015.

[6] Resnick Mitchell. Scratch: Programming for all. *Communications of the ACM*, 52 (11): pp. 60-67, 2009.

[7] X. Molero-Prieto, A. Veloso-Padilla. El Museo de Informática de la UPV como puerta de entrada a las enseñanzas técnicas universitarias: compromiso con la sociedad en sus actividades didácticas. *Congreso Universitario de Innovación Educativa en las Enseñanzas Técnicas*, ISBN: 978-84-09-02970-9, 2019.

[8] Dirk Huylebrouck. *Africa and Mathematics From Colonial Findings Back to the Ishango Rods*. Springer Nature Switzerland AG, Suiza, 2019.

[9] César Vidal Manzanares. *La sabiduria del antiguo Egipto*. Alianza Editorial, S.A, Madrid, 1994.

[10] K. Ríbnikov. *Historia de las Matemáticas*. Editorial Mir Moscú, Moscú, Primera edició 1987.

[11] Georges Ifrah. *Historia universal de las cifras*. Editorial Espasa, Pozuelo de Alarcón, 2008.

[12] Michael R Williams. *History of computing technology*. The Institute of Electrical and Electronics Engineers, Inc, California, Second edition, 1997.

[13] Michael Badger. *Scratch 2.0 Beginner's Guide*. Packt Publishing Ltd, Birmingham UK, Second edition, 2014.

[14] Luis Fernandez, Una Breve Historia del Ábaco Consultado en https://www.ee.ryerson.ca/~elf/abacus/espanol/history.html el 12 de febrero del 2020.

[15] Yves Serra, La machine arithmétique de Leibniz Consultado en https://journals.openedition.org/bibnum/551 el 23 de febrero del 2020.

[16] Bruce Collier, The Little Engines that Could've: The Calculating Machines of Charles Babbage Consultado en http://robroy.dyndns.info/collier/index.html el 25 de febrero del 2020.

[17] Thomas de Colmar Charles-Xavier, Patent Nº 8282 - 25th april 1849 Consultado en http://www.arithmometre.org/Brevets/PageBrevet1849EN.html el 27 de febrero del 2020.

[18] John Wolff, Pin-wheel Calculators Consultado en http://www.johnwolff.id.au/calculators/pinwheel/pinwheel.htm el 2 de marzo del 2020.

[19] John Wolff, "FacitÇalculators Consultado en http://www.johnwolff.id.au/calculators/Facit/Facit.htm el 7 de marzo del 2020.

[20] Harald Schmid, Vom Modell T zur 1051 Die 10 Tasten Universalrechenmaschinen von FACIT Consultado en http://www.rechenautomat.de/Facit/index.php el 16 de marzo del 2020.

[21] Elvia R. De Gracia C., La importancia de la utilización de Scratch en la Educación Consultado en http://eduteka.icesi.edu.co/gp/upload/c3d23e442dbd2a36cf18b86b1291aba5.pdf el 25 de marzo del 2020.

APÉNDICE A
Implementación de la lógica

La implementación realizada en Scratch para los bloques Rotor y Motor son relativamente largos, debido a que el lenguaje de programación es de alto nivel con pocas opciones de simplificación de código. Por este motivo se decidió mostrar al lector de otra manera la estructura, desarrollando en pseudocódigo tipo JavaScript de la algoritmia creada. Cabe añadir que el bloque Motor tiene agregado una serie de comentarios con respecto a cada parte del código mas relevante, ofreciendo como objetivo el entendimiento del algoritmo cuando realiza una opción.

- **Pseudocódigo del bloque Rotor:**

```
1
2     function Rotor(u){
3         var res;
4         for(k = t; k < p+t; k++)
5         {
6             res = RW[k]+u*EW[k+9-(p+t)];
7             RW[k] = (10+res)%10;
8             RZ[k] = res<0||res>9?1:0;
9         }
10        for(k = t+1; k < 13; k++)
11        {
12            res = RW[k]+u*RZ[k-1];
13            RW[k] = (10+res)%10;
14            RZ[k]+= res<0||res>9?1:0;
15        }
16        d = RZ[12]; Null(RZ); RWerk();
17    }
```

- **Pseudocódigo del bloque Motor:**

```
1
2 function Motor(){
3
4         if (U != 0)
5         {
6             Rotor(U);
7             if (r != 0)
8                 Quot(r*U);
9         }
```

```
10
11          if (M == 1){ //Multiplicacion automatica
12              Z = (Z-r*U) %10; // Contador multiplicador
13              if (Z == 0){
14                  if (m > 0){ LTab(); a--;} // Tabular
15                  m++;
16                  if (a < 0){
17                      M = 0; m = 0;
18                      U = 0; V = 1;
19                  }else{ //Escanear el multiplicador
20                      var res = z + MW[8-a];
21                      if (m < 8){
22                          Z = res %10;
23                          z = res > 5? 1 : 0;
24                          U = (Z==0) ? 0 : (Z > 5) ? -r : r;
25                      }else{ //Caso especial 8 digitos
26                          Z = res;
27                          z = 0;
28                          U = (Z==0) ? 0 : 1;
29                          }
30                      }
31              }
32          }else{
33              if (U != 0 && S != 0){ //Sub-Stop
34                  U = 0; S = 0; ZNull(1);
35                  if (r == 0) r = -1;
36              }
37              else{
38              if (V <= 1){ //Semi-automatico
39                  if (U == 0 && D != 0 && H < 0){ //Comienza
                        division automatica
40                      V = 2; d = 0;}
41                  if (U != 0 && D == 0 && H >= 0){ //Tabulador
                        automatico
42                      if (2*H+A > 0){LTab(); if (K==1){s=8; v=0;}}
43                      if (2*H+A < 0){RTab(); if (K==1){s=8; v=0;}}
44                  }
45                  U = D;
46                  if (U != 0 && V == 0) V = 1;
47
48              }else{ // Division automatica
49                  if (H < 0 && d > 0){ //Impulso de la division
50                      U = -U; d = 0;
51                      if (U == D) {U = 0; V = 1;} //Tecla presionada
                            ?
52                      if (q == 1 || t == 0) U = U > 0? 1 : 0; //
                            Detener la operacion
53                      if (U == 0) {D = 0; V = 1;} //Fin division
                            automatica
54                          RTab();
55                      }
56                  }
57              }
58          }
59      }
```